THE UNIVERSITY OF MICHIGAN

COLLEGE OF ENGINEERING
Department of Atmospheric, Oceanic and Space Sciences
Space Physics Research Laboratory

Advanced Langmuir Probe

FINAL REPORT

Prepared on behalf of the project by:

N. R. Voronka
B. P. Block
G. R. Carignan

Under Contract with:
National Aeronautics and Space Administration
Goddard Space Flight Center
Grant No. NAG5-419
Greenbelt Maryland 20771

Final Project Report

Advanced Langmuir Probe

Grant No. NAG5-419

Prepared by:

N. R. Voronka
B. P. Block
G. R. Carignan

Space Physics Research Laboratory
University of Michigan
Ann Arbor, Michigan 48109-2143

TABLE OF CONTENTS

Final Project Report

Advanced Langmuir Probe

Grant No. NAG5-419

Prepared by:

N. R. Voronka
B. P. Block
G. R. Carignan

Space Physics Research Laboratory
University of Michigan
Ann Arbor, Michigan 48109-2143

TABLE OF CONTENTS

Introduction

For more that two decades, the staff of the Space Physics Research Laboratory (SPRL) has collaborated with the Goddard Space Flight Center (GSFC) in the design and implementation of Langmuir Probes (LP). This program of probe development under the direction of Larry Brace of GSFC has evolved methodically with innovations to: improve measurement precision, increase the speed of measurement, and reduce the weight, size, power consumption and data rate of the instrument.

During the course of probe development of the Pioneer Venus and Dynamics Explorer systems, it was determined that the speed of the electrometer, which is the basic detector of the LP, was the rate determining component of the instrument. Under NASA Contract NAS5-27305 an in depth study of electrometer response was done, which resulted in a list of recommendations to improve the speed of the electrometer without sacrificing any precision. Under this contract, these improvements were to be implemented, and the characteristics of the modified electrometer were to be measured. To further increase the speed of measurements, the measurement algorithm was to be modified. The implementation of these changes was to be realized in a configuration of reduced size, weight, and power. All of these improvements characterize the Advanced Langmuir Probe (ALP).

Summary of Project Activities

The report by J. Dittmar and A. Macnee proposed techniques to improve the speed and accuracy of the LP MK-2 pre-amplifier. Based on these recommendations, a laboratory model was constructed using components and techniques that are directly applicable to a flight program. The amplifier's performance was studied and it was concluded that the new amplifier had improved step-input settling times, recovery from range changes and recovery from saturation. The worst case settling time of the pre-amplifier was improved by nearly an order of magnitude.

The next phase of this study was the development and implementation of the improved measurement algorithm. This five-point algorithm was a significant improvement over the algorithms used on Pioneer Venus and Dynamics explorer. Through the use of a microprocessor, this algorithm was implemented to reduce measurement time and required data rate, thereby increasing temporal resolution of the instrument. A dedicated processor system based on the Harris 80C86 was developed which produces the voltage applied to the probe and measures the resultant probe current.

The measurement algorithms were implemented and tested under a real-time executive (Ready Systems VRTX). Also, application software was written to allow the graphical display of acquired data. The graphic display of the V-A function allowed easy evaluation of the five-point algorithm and the accuracy of the plasma simulators.

The electrometers designed and implemented at SPRL are floating atop the applied voltage to simplify the circuitry that generates this voltage. However, this exacerbates the problem of switching power converter noise in the measurement. In this study, the problem was solved by increasing the frequency of the power converter so that its fundamental frequency was outside the response bandwidth of the current amplifier. When simulated plasma measurements were made, dramatic improvements resulted.

The last phase of this study began the development of flight electronics to test the above improvements on a sounding rocket. In addition to the ALP, these electronics will support an additional instrument. The Solar Flux Monitor (SFM) is an instrument that will be used to measure Extreme Ultraviolet Radiation, and is being developed by Dr. Walter Hoegy and Larry Brace of GSFC. Since measurements with the SFM require an applied voltage and produce a current output in similiar ranges as the ALP, the measurement electronics for the SFM will be derived closely from the work performed during this study. This effort is being continued on NASA Contract NAG5-1691.

BLOCK

UNIVERSITY OF MICHIGAN
SPACE PHYSICS RESEARCH LABORATORY
July 28, 1983

MEMO TO: G. R. Carignan

FROM: James Dittmar and Alan Macnee

SUBJECT: An Analytic and Experimental Study of Variable Gain Langmuir Probe Amplifier Circuits

SUMMARY

The dynamic response of the MK-2 version of the Langmuir probe amplifier has been studied. The settling time of the step response is increased by:

1) stray node-to-ground capacitance at series connections between high value feedback resistors;

2) input capacitances due to the input cable, FET switches, and input source follower.

Step response measurements show that the MK-2 circuit was inadequately compensated for stray and input capacitances. On the highest gain setting, the response was underdamped, showing 96 percent overshoot, and a settling time of 1 millisecond.

The stray node-to-ground capacitances can be reduced to tolerable levels by elevating the string of feedback resistors above the printed circuit board. This isolates the series connections from ground paths. Under these conditions, a frequency response having Butterworth characteristics was achieved, with no compensating components. The 90 percent rise time was about 200 microseconds at the highest gain setting. (The input cable capacitance was 12 picofarads.)

A new feedback network was considered, with promising results. The design uses resistances having much lower nominal values, thereby minimizing the affect of stray capacitances. The measured rise time at the highest gain setting was 200 microseconds, which is the same result achieved for the elevated resistor case. (The same printed circuit board was used, and the input cable capacitance was 12 picofarads.)

Still faster settling times can be achieved by using an operational amplifier having a higher gain-bandwidth product. The rise time was reduced to 77 microseconds by substituting an operational amplifier having a 6.0 MHz gain-bandwidth product into the redesigned circuit.

The basic MK-2 amplifier is shown in Figure 1, and the step response measurements made are summarized in Table 1. Two versions of the new feeback design are shown in Figures 2 and 3, and the measurements made are summarized in Table 2. The analytical expressions for selecting the compensation components, and for predicting the step response rise times are summarized in Table 3.

ANALYSIS

The basic Langmuir probe amplifier is shown in Figure 1. Using feedback analysis, the forward transfer impedance, z_{ft}, is approximately:

$$z_{ft} = \frac{v_o}{i_{in}} = \frac{\dfrac{-\alpha R_f}{(\frac{s}{a} + 1)(\frac{s}{g} + 1)}}{1 + \dfrac{\alpha R_f (\frac{s}{f} + 1)(\frac{s}{x} + 1)}{(\frac{s}{a} + 1)(\frac{s}{g} + 1)(\frac{s}{c} + 1) R_f}}$$

where α is the open loop voltage gain of the operational amplifier, and

$$\frac{1}{g} = R_f (C_{in} + C_f)$$

$$C_{in} = C_{amplifier} + C_{cable}$$

$$\frac{1}{f} = R_f C_f$$

$$\frac{1}{c} = R_c (C_x + C_c)$$

$$\frac{1}{a} = \frac{\alpha}{2\pi\ GB}$$

and, $$\frac{1}{x} = R_c C_x$$

The gain-bandwidth product of the operational amplifier is GB, and C_{in} is the total input capacitance from the input cable, FET switches, and input source follower.

In the MK-2 circuit, $\frac{1}{c}$ is adjusted to equal $\frac{1}{f}$. Cancelling terms, and simplifying:

$$z_{ft} = \frac{-\alpha R_f \, ag}{s^2 + (a + g + \frac{\alpha ag}{x})\, s + \alpha ag}$$

The $\frac{1}{x}$ term can be adjusted to compensate for input capacitance, providing $c_x << c_c$ so that the $\frac{1}{f}$ term will still cancel the $\frac{1}{c}$ term. Conversely, if the composite FET - operational amplifier is unstable as a voltage follower, C_x must be large enough to prevent oscillation on the low gain settings.

For a step response having minimum rise time and no overshoot, (Thomson response), the following relationship exists:

$$s^2 + (a + g + \frac{\alpha ag}{x})\, s + \alpha ag = s^2 + 2\sqrt{3}\, W_o s + 4W_o^2$$

where W_o is the imaginary part of the complex conjugate roots. Equating terms, and solving for the compensation time constant, $\frac{1}{x}$,

$$\frac{1}{x} = \frac{\sqrt{3\ ag - (a + g)}}{\alpha ag} \simeq \sqrt{\frac{3}{\alpha ag}}$$

For a Butterworth response, the result is:

$$\frac{1}{x} \simeq \sqrt{\frac{2}{\alpha ag}}$$

The most favorable response is obtained by a design in between the Thomson and Butterworth designs.

The above analysis neglects stray node-to-ground capacitances, so it cannot be strictly applied to the MK-2 circuit. (It can be applied if the feedback resistors are isolated from ground paths, so that the stray node-to-ground capacitances are reduced to much lower proportions.) However, it can be used to solve for a minimum value for the compensating capacitor, C_x. For example, using the Butterworth criterion, if C_{in} = 26 pf, C_c = 18 nf, C_f = 0.1 pf, R_f = 1 giga-ohm, GB = 1.35MHz, and R_c = 11.3 k-ohms, then $\frac{1}{x}$ = 78.4 microseconds, and $C_x = 1/xR_c$ = 6.94 nf. The inequality, $C_x << C_c$, is not met, and the circuit cannot be completely compensated for input capacitance by adjusting C_x.

NEW DESIGN

Alternatives to the MK-2 feedback circuit are shown in Figures 2 and 3. Because the feedback resistors are lower in value, and lumped, the problem of stray capacitance is greatly reduced. The circuit can be compensated for input capacitance by C_f in parallel with R_f, or by C_x in parallel with R_c. For the lower gain settings, C_f should be used to insure the stability of the amplifier. For the highest gain setting, C_x is preferred, because its value will be more practical, and easier to trim, than the corresponding C_f.

The analysis for this circuit is similar to the analysis for the MK-2 circuit. The forward transfer impedance, z_{ft}, is approximately:

$$z_{ft} = \frac{\dfrac{-\alpha R_f}{(\frac{S}{a}+1)(\frac{S}{g}+1)}}{1 + \dfrac{\alpha R_f (\frac{S}{\Gamma}+1)}{(\frac{S}{a}+1)(\frac{S}{g}+1)\ R_f\ k\ (\frac{S}{c}+1)}}$$

where K is the divider ratio, (e.g., $K = (R_1 + R_c)/R_1$), and $1/c = R_c R_1 Cx/(R_1 + R_c)$. For large K, the 1/C term can be neglected. The analysis also neglects the output resistance of the operational amplifier, so R_1, should be much greater than the output resistance if C_x is used. The $1/\Gamma$ term is either $C_f R_f$ or $C_x R_c$, depending where the compensating capacitor is placed.

For a Thomson response, the compensation time constant, $\frac{1}{\Gamma}$, is:

$$\frac{1}{\Gamma} = \frac{\sqrt{3\alpha ag - (a+g)}}{\alpha ag} \approx \sqrt{\frac{3k}{\alpha ag}}$$

The unit step response, for t>0, is:

$$H(t) = 1 - 2e^{-\sqrt{3}\, w_o t} \sin(w_o t + 30^\circ)$$

where $w_o = \frac{1}{2}\sqrt{\frac{\alpha ag}{k}}$

For a Butterworth response, the results are:

$$\frac{1}{\Gamma} \approx \sqrt{\frac{2k}{\alpha ag}}$$

$$H(t) = 1 - \sqrt{2}\, e^{-w_o t} \sin(w_o t + 45^\circ), \text{ for } t>0$$

$$\text{and, } w_o = \sqrt{\frac{\alpha ag}{2k}}$$

For equal real poles, the denominator of x_{ft} is a perfect square. The results are:

$$\frac{1}{\Gamma} = 2\sqrt{\frac{k}{\alpha ag}}$$

$$H(t) = 1 - e^{-\sigma t}[1 + 2t], \text{ for } t>0$$

$$\text{and, } \sigma = \sqrt{\frac{\alpha ag}{k}}.$$

The most favorable response is obtained by a design in between the Butterworth and Thomson designs. The three types of response are compared in Figure 4.

The divider ratio should be large enough to insure small time constants due to the stray capacitances in the feedback network. The feedback resistors must also have low enough values to be lumped. Conversely, the divider ratio must be low enough to keep the circuit desensitized with respect to the operational amplifier open loop voltage gain. The analytical expressions needed to select the compensation components and to predict the step response rise times follow directly from the above analysis. They are summarized in Table 3.

NOISE

The root mean square output noise voltage due to the feedback resistor in the MK-2 circuit is $V_{nm} = i_{nm}R_{fM}$ where i_n is the root mean square noise current due to R_{fM}. The noise current is proportional to $1/\sqrt{R_{fM}}$, so the noise voltage is proportional to $\sqrt{R_{fM}}$. For the redesigned circuit, the output noise voltage is $V_{nr} = i_{nr}kR_{fr}$, which is proportional to $k\sqrt{R_{fr}}$. The ratio of V_{nr} to V_{nm} is $k\sqrt{\frac{R_{fr}}{R_{fM}}}$, and $R_{fM} = k\,R_{fr}$. Therefore, the low frequency output noise voltage due to the feedback resistor in the redesigned circuit is $\sqrt{K}$ times higher than for its counterpart in the MK-2 circuit.

RESULTS

Step response measurements were made on the MK-2 circuit, and on several variations of the new feedback design. Each circuit was breadboarded on the MK-2 printed circuit board, and housed in an aluminum chassis. The amplifier input was coupled through a 3.3 picofarad capacitor to a triangle wave generator with a 50 ohm source impedance. The triangle wave is differentiated by the coupling capacitor to give a square wave input current, the peak to peak value of which is (3.3 pf) (2) $(\Delta V_{in}/\Delta t)$. The capacitance of the input cable, measured from the amplifier input end, was 12 picofarads.

Ciran analysis of the MK-2 feedback network shows that the stray node-to-ground capacitances can have a dominant effect on the amplifier response. There is evidence of this in the measured step response on the highest gain setting: Insertion or removal of the 1 nf compensating capacitor causes negligible change; and, in both cases, the response was underdamped, showing 96 percent overshoot, and a settling time of 1 millisecond. Figure 5 shows the step repsonse on the highest gain setting with, and without, the 1 nf capacitor.

When the 200 megohm resistors were mounted one inch above the printed circuit board, a response having Butterworth characteristics was achieved with $C_f = C_c = C_x = 0$. This shows the dramatic consequences of reducing the stray node-to-ground capacitances. The absence of ringing owes to the coincidental compensation of input capacitance by the stray capacitance that

shunts the total feedback resistance. High overshoot and settling times were easily induced by adding small stray capacitances at the series connections of the feedback resistors. This confirms the importance of these capacitances in the MK-2 layout. The photographs in Figure 5 show the MK-2 response on the highest gain setting with the feedback resistors mounted above, and on, the circuit board.

The amplifier was unstable in the lowest two range settings with the 1 nf capacitor removed. The composite FET - operational amplifier was also unstable as a voltage follower. The operational amplifier alone as a voltage follower was stable, but showed increased overshoot and settling time with load capacitance. Careful measurements were made to characterize the ZN5906 and the HA2700, so that the circuit model would predict this instability. The models developed, and graphs comparing the actual and simulated step responses are shown in Figures 6 and 7.

The first new design tested is shown in Figure 2. The measured rise times for the three highest gain settings were 200, 67, and 20 microseconds. The feedback capacitors required to compensate for the total input capacitance were 7.6, 1.4, and 27 picofarads, respectively. Compensation was attempted by placing a capacitor across the top divider resistor, instead of across the feedback resistor, but the lowest two gain settings oscillated. This is explained by the instability of the composite voltage follower.

The photograph in Figure 8 shows the uncompensated response for this circuit for three values of input capacitance, and on the highest gain setting. In this case, the forward transfer impedance is dominated by the $R_f C_{in}$ time constant, and by the lowest pole of the operational amplifier. The amplifier input capacitance can be obtained from the ringing frequency, w_o. The expression is:

$$C_{amplifier} = \left\{ \frac{1}{2R_f\left(\frac{a\alpha}{k} - \sqrt{\left(\frac{a\alpha}{k}\right)^2 - w_o^2}\right)} \right\} - C_{cable}$$

Averaging over the three measurements, the amplifier input capacitance was calculated to be 13.7 picofarads.

The circuit shown in Figure 3 combines two compensaton schemes. Compensating the highest gain setting at the divider makes the response easier to adjust. Where C_{f4} could be less than 1 picofarad, C_x will be many times higher. The other ranges were compensated individually with capacitors shunting the feedback resistors. This insures the stability of the lowest ranges. Note that R_1 should be much greater than the output resistance of the operational amplifier for the best results. The measured rise times were 200, 52, 19, and 7.6 microseconds for the highest to lowest gain settings. The rise times with 48.5 picofarads added to the input were 340, 92, 25, and 9.6 microseconds, respectively.

The analysis predicts that the compensated response for this circuit should not depend on the divider ratio of the highest

gain setting. This result was demonstrated by building a circuit for which R_f = 35 megohms, and K = 28.6. The measured rise times were 200 and 340 microseconds, for no added input capacitance, and for 48.5 picofarads added to the input. This is the same result obtained above, for R_f = 10 megohms, and k = 101. For R_f = 200 megohms, K = 5, the rise times were 230 and 360 microseconds, for no added input capacitance, and for 48.5 picofarads added to the input. The slight increase is because the divider ratio is low, and the pole associated with R_c, R_1, and C_x is closer to the compensation zero.

A PMI type OP-15 operational amplifier was substituted for the Harris HA2700 to test the dependence on gain-bandwidth product. The OP-15 has a gain-bandwidth of about 6.0 MHz. For the two cases above, the rise times were 90 and 150 microseconds for R_f =30 megohms, and 77.4 and 185 microseconds for R_f = 200 megohms, and for the respective values of input capacitance. Comparing the OP-15 results with the HA2700 results for the circuit with the higher divider ratio, there is a factor of 2.2 improvement in rise time. The analysis predicts that the rise time will depend on the square root of the gain-bandwidth product. Checking this result, 6 MHz/1.35 MHz = 4.84, and $(2.2)^2$ = 4.44, and the theory is corroborated.

The circuit in Figure 3 could be designed so that the compensation of the highest gain setting would exactly compensate the next lower gain setting. In other words for proper choice of R_{f4} and C_x, C_{f3} will not be needed. Experimentally, the next to

the highest gain setting was just slightly overcompensated with C_{f3} = 0, R_{f4} = 35 megohms, and C_x adjusted for a Thomson response on the highest gain setting. (The cable capacitance was 12 pf.)

CONCLUSIONS

The settling time of the MK-2 circuit is increased by stray node-to-ground capacitances in the feedback loop, and by input capacitance. The circuit does not adequately compensate for the effects of these capacitances. The stray node-to-ground capacitances can be reduced to acceptable proportions by mounting the string of high value feedback resistors above the printed circuit board. This would decrease the stray capacitances by isolating the series connections from ground paths. The technique would require considerable layout and construction care, and the circuit would still be undercompensated for input capacitance.

The settling time of the amplifier can be reduced by using a feedback network such as the one shown in Figure 3. The feedback resistors are much lower in value than those used in the MK-2 circuit, and they can be lumped. The resistance values are low enough to reduce the effects of stray shunt capacitances to acceptable proportions. Lumping eliminates the stray node-to-ground capacitances, and is equivalent to mounting the feedback resistances above the board in the MK-2 circuit. The layout and construction of the new circuit is not as critical. The circuit can be compensated for input capacitance using components that are practical in value. The compensation can be tailored to each gain setting, and the number of components required is low. The circuits tested worked well in all of the above aspects.

Still faster settling times can be achieved by using an operational amplifier with higher gain-bandwidth product in the redesigned circuit. The compensated rise time is proportional to the reciprocal of the square root of the gain-band width product.

JD:scg

Table 1

Conditions						
C_f	0	in	in	in	0	in
C_c	0	0	18 nf	18 nf	0	0
C_x	0	0	0	Inf	0	0
R_f	on p.c. board	on board	on board	on board	elevated	elevated
$\frac{V_o}{i_{in}} = Z_{ft}$	Settling or rise time (90%), and % overshoot					
250K	oscillates	oscillates	oscillates			
4.17M	60 µS 54%	oscillates	oscillates			
66M	300 µS 60%	480 µS 0%	1.2 mS 100%			
1000M	2mS, 176%	1.25mS 20%	1mS 100%	1mS 96%	200µS 4%	500µS 0%

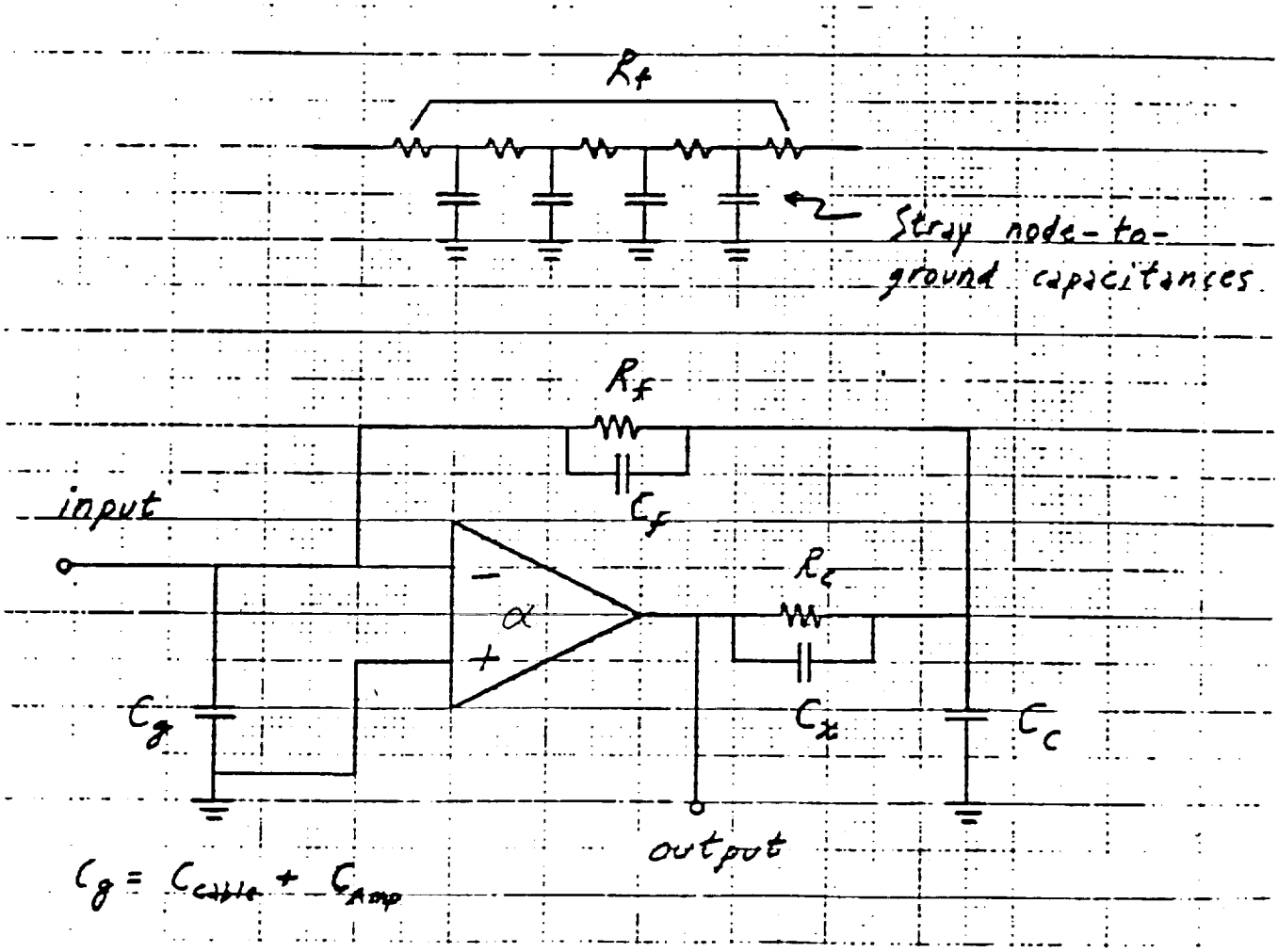


Figure 1 Basic MK-2 circuit.

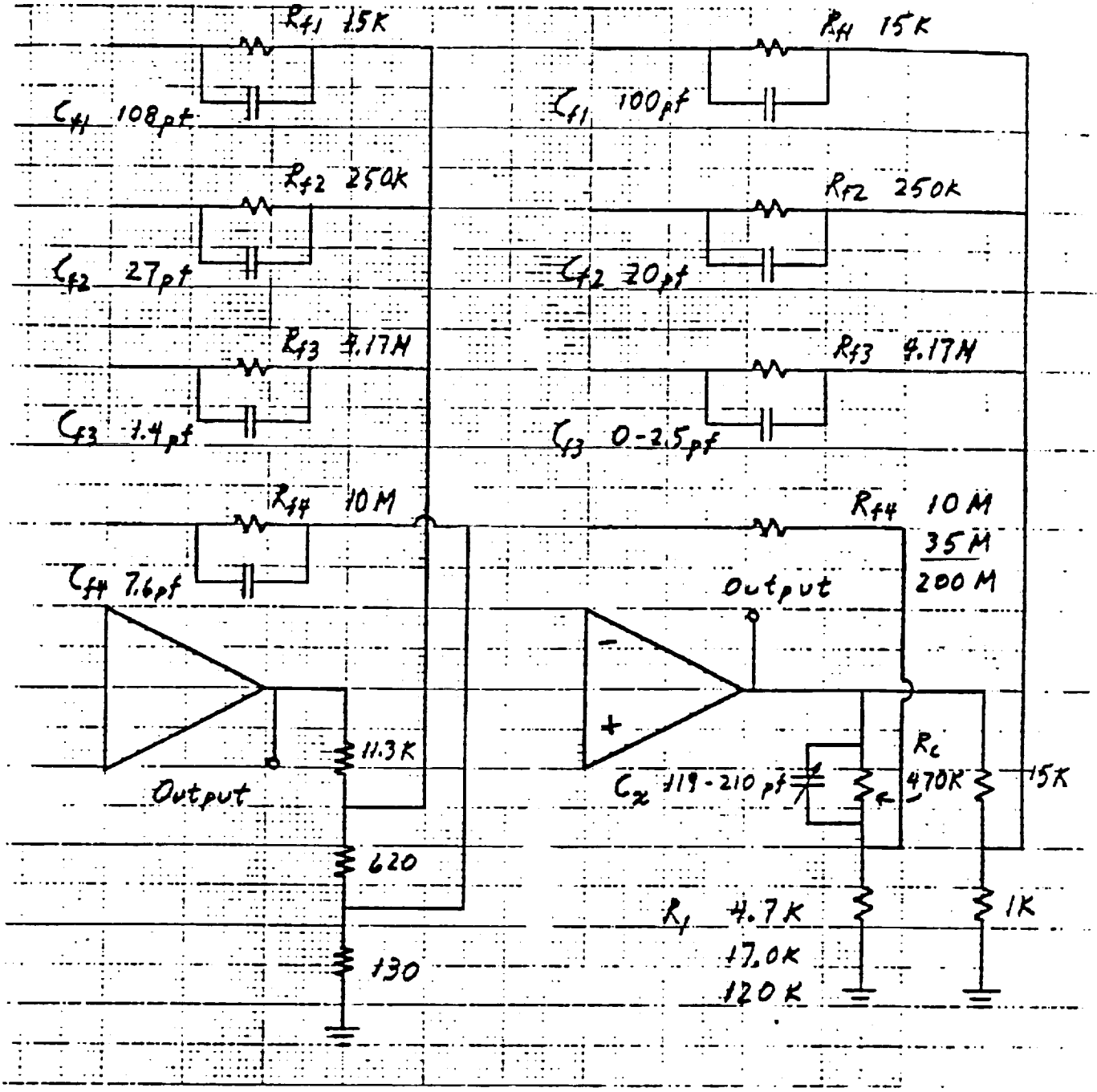


Figure 2 First new circuit.

Figure 3 Second new circuit.

Circuit Conditions	New circuit step response measurements											
Circuit figure	2	3	3	3		3		3		3		
R_{f4} (megohms)	10	10	10	35		35		200		200		
Input cable capacitance (pf)	12	12	60.5	12		60.5		12		60.5		
Operational Amplifier	HA 2700	HA 2700	HA 2700	HA 2700	PMI OP-15	HA 2700	PMI OP-15	HA 2700	PMI OP-15	HA 2700	PMI OP-15	
$\frac{V_{OUT}}{i_{IN}}$	Compensated step response rise times (90%) in μsec.											
250K		7.6	9.6									
4.17M	20	19	25									
66 M	67	52	92									
1000M	200	200	340	200	90	340	150	230	77	360	185	

Table 2 Step response measurements for circuits shown in figures 2 and 3.

Response type	C_x	C_{fj} - total parallel comb. $j = 1, 2, 3$	σ	ω_o	Step response $t > 0$	Rise time (90%)
Butterworth	$\frac{1}{R_c}\sqrt{\frac{2k}{\alpha a g}}$ $= \frac{1}{R_c}\sqrt{\frac{K_4 R_{f4} C_{IN}}{\pi GB}}$	$T + \sqrt{T^2 + 2T C_{IN}}$ $T_B = \frac{k_j}{R_{fj}\,\alpha a}$	ω_o	$\sqrt{\frac{\alpha a g}{2k}}$	$1 - \sqrt{2}e^{-\sigma t} \times \sin(\omega_o t + 45^\circ)$	$\frac{1.87}{\omega_o}$
Thomson	$\frac{1}{R_c}\sqrt{\frac{3k}{\alpha a g}}$ $= \frac{1}{R_c}\sqrt{\frac{3K_4 R_{f4} C_{IN}}{2\pi GB}}$	$T = \frac{3k_j}{2R_{fj}\,\alpha a}$	$\sqrt{3}\,\omega_o$	$\frac{1}{2}\sqrt{\frac{\alpha a g}{k}}$	$1 - 2e^{-\sigma t} \times \sin(\omega_o t + 30^\circ)$	$\frac{1.62}{\omega_o}$
Perfect square	$\frac{2}{R_c}\sqrt{\frac{k}{\alpha a g}}$ $= \frac{1}{R_c}\sqrt{\frac{2K_4 R_{f4} C_{IN}}{\pi GB}}$	$T = \frac{2k_j}{R_{fj}\,\alpha a}$	$\sqrt{\frac{\alpha a g}{k}}$	0	$1 - e^{-\sigma t}(1 + 2t)$	$\frac{3.88}{\sigma}$

Table 3 Analytical expressions for circuits in figures 2 and 3.

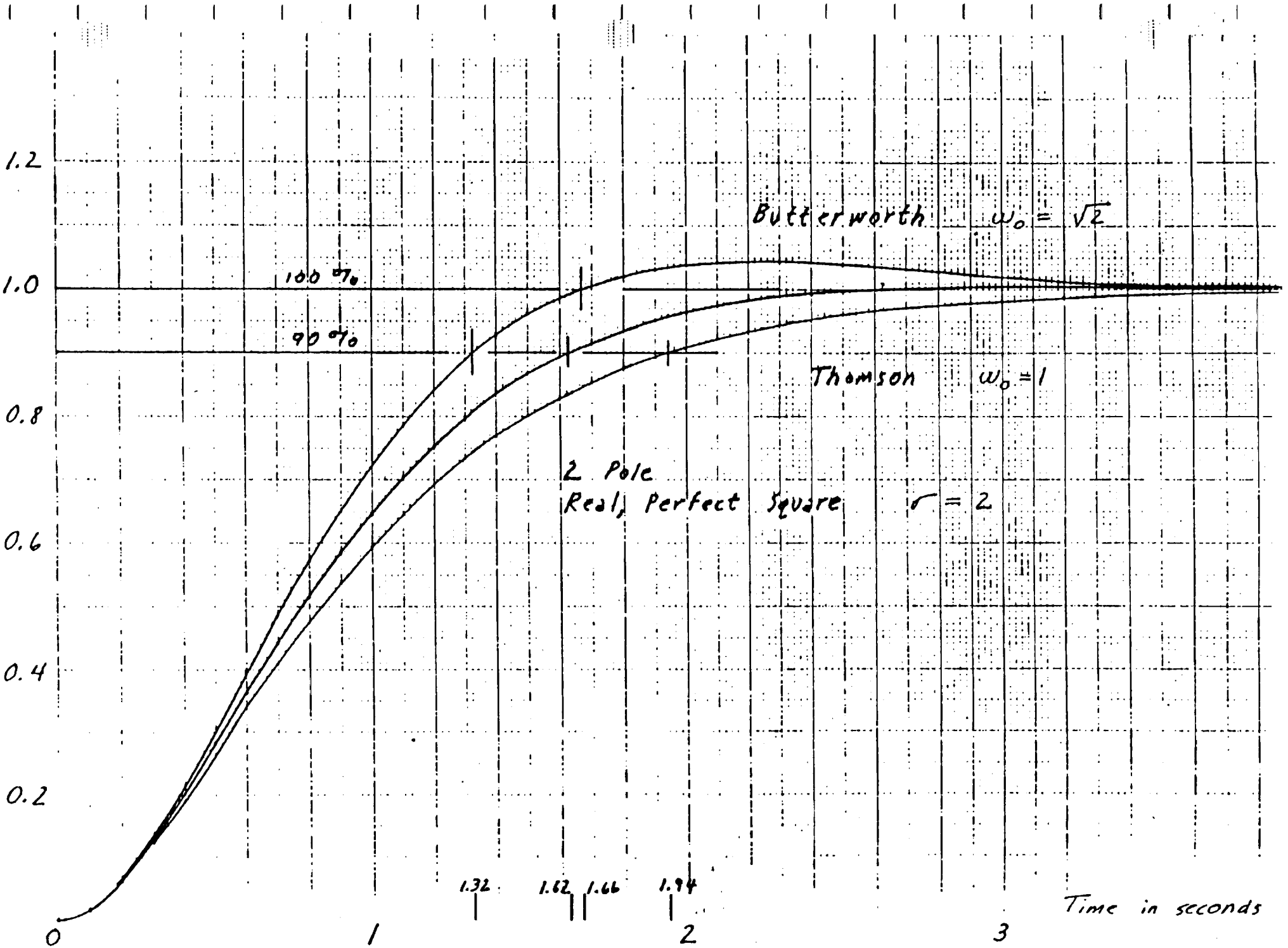


Figure 4 Normalized Butterworth, Thomson, and perfect square step responses.

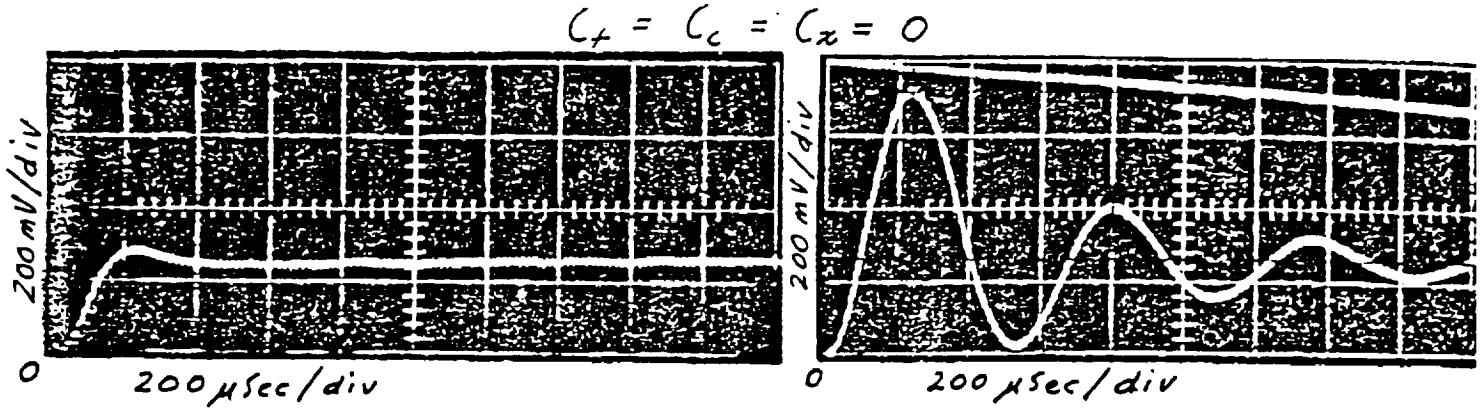


Figure 5a MK-2 step response with feedback resistors mounted above the printed circuit board, (left), and on the board, (right), on the highest gain setting.

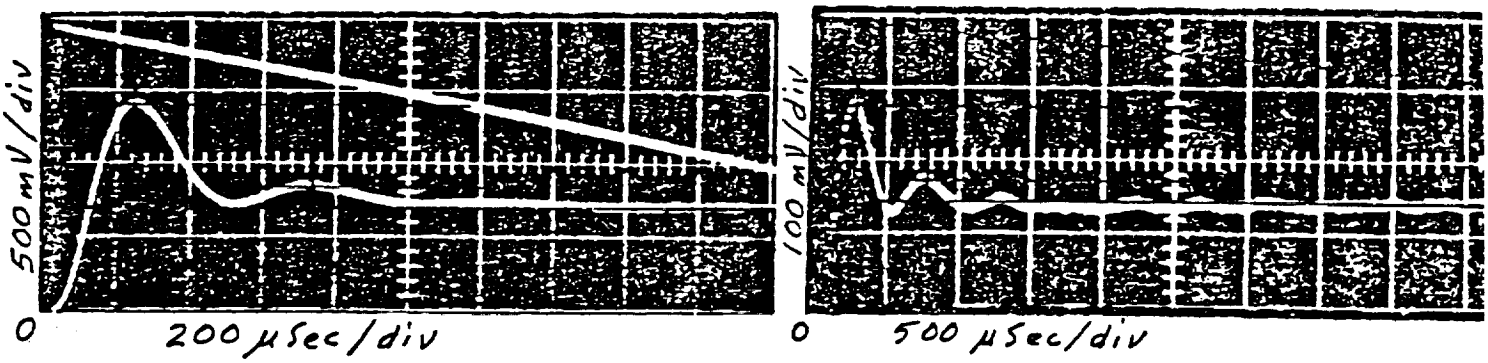


Figure 5b MK-2 step response without C_X, (left), and with C_X, (right), on the highest gain setting.

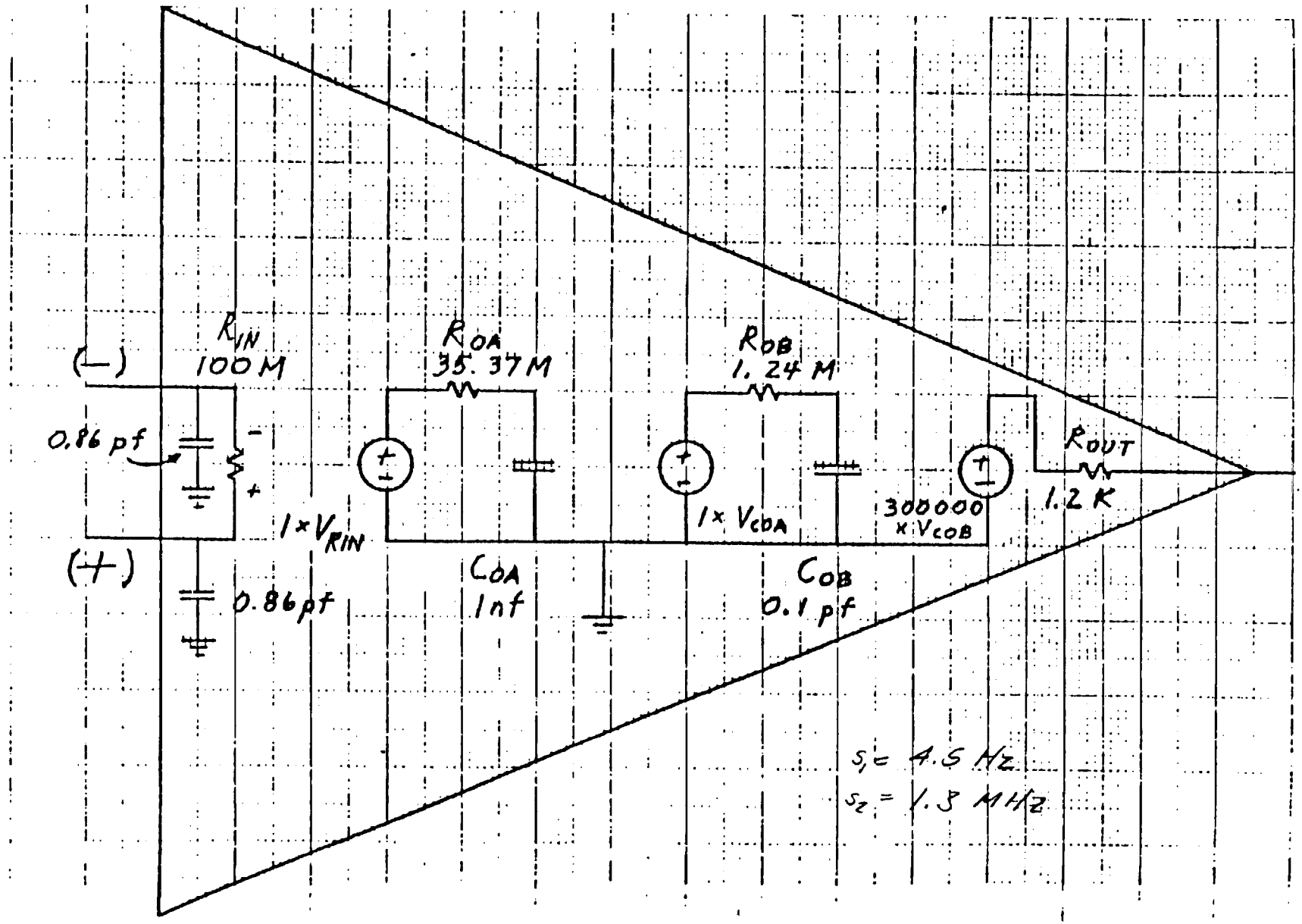


Figure 6a HA2700 operational amplifier model.

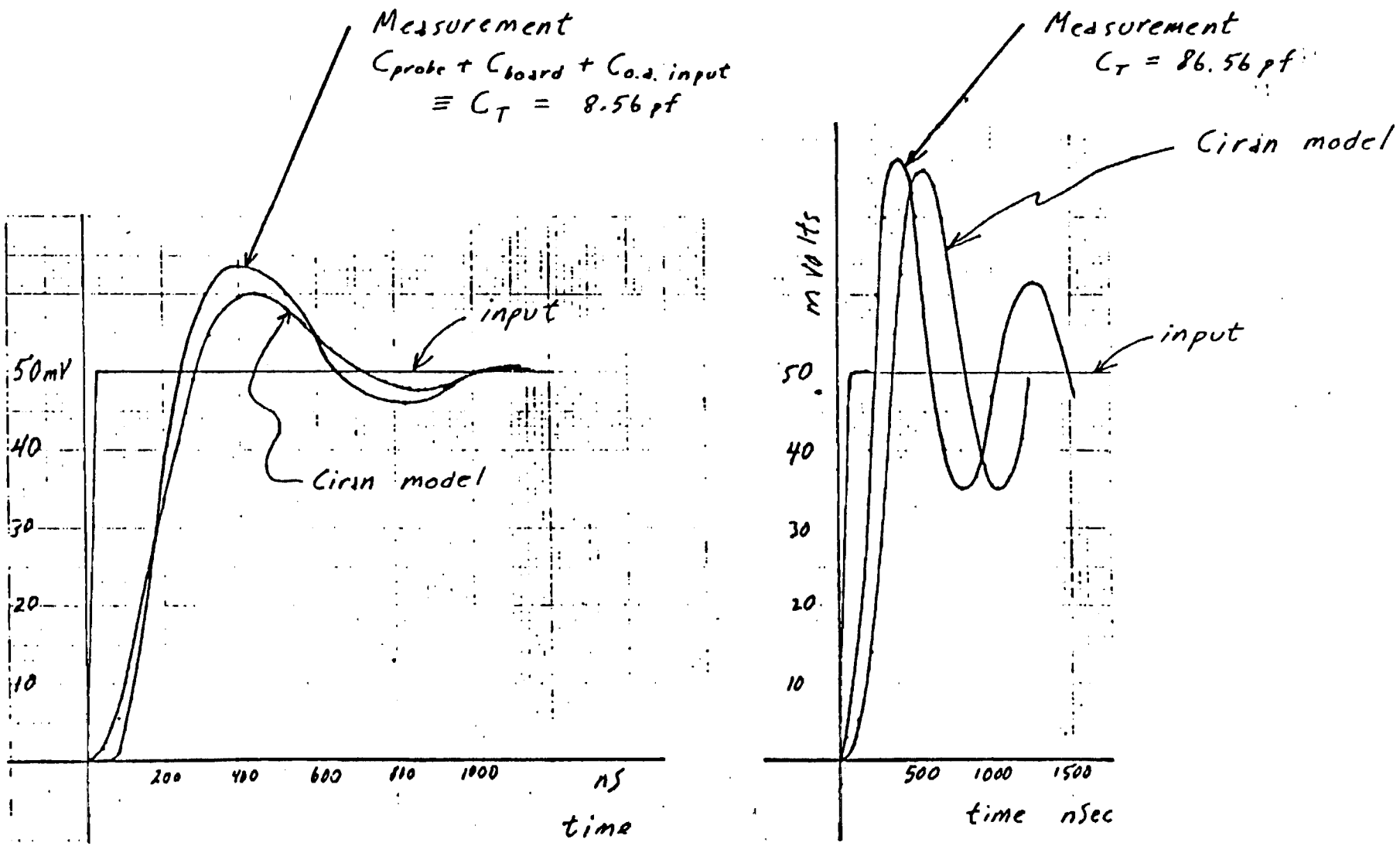
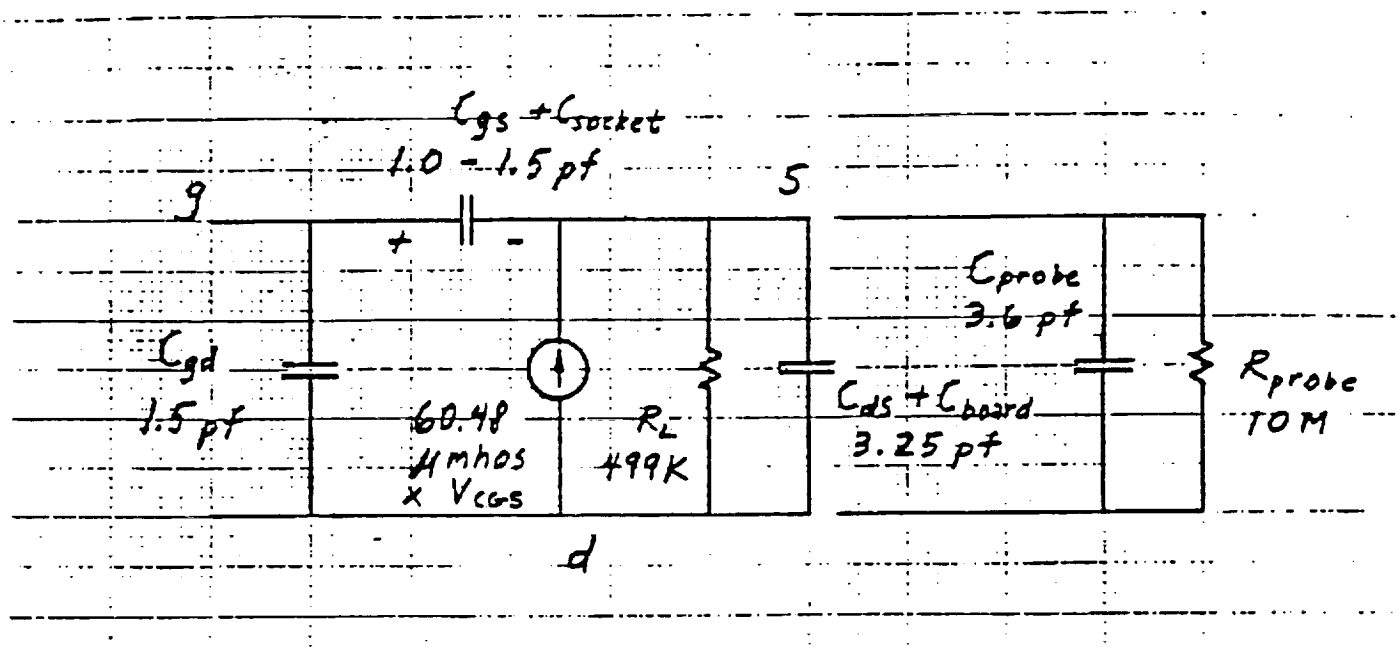


Figure 6b Predicted and measured operational amplifier voltage follower step response.



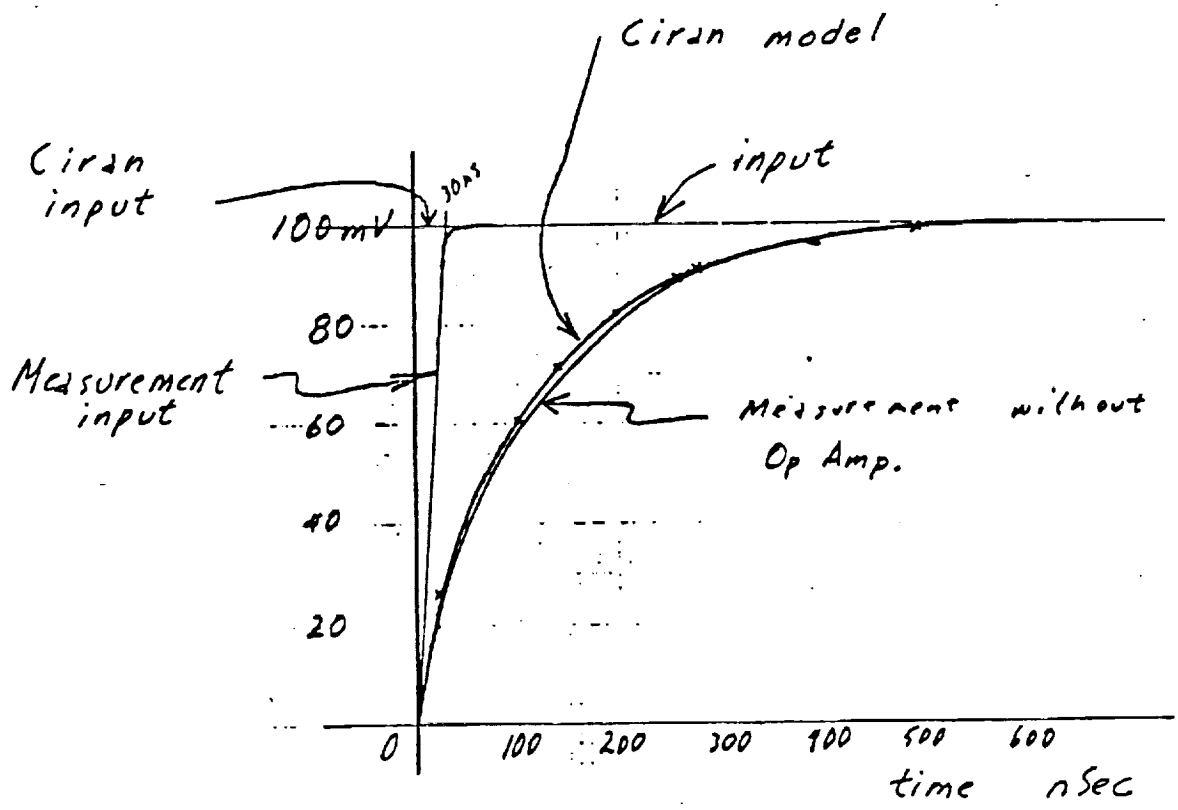


Figure 7 2N5906 FET model, and predicted and measured source follower step response.

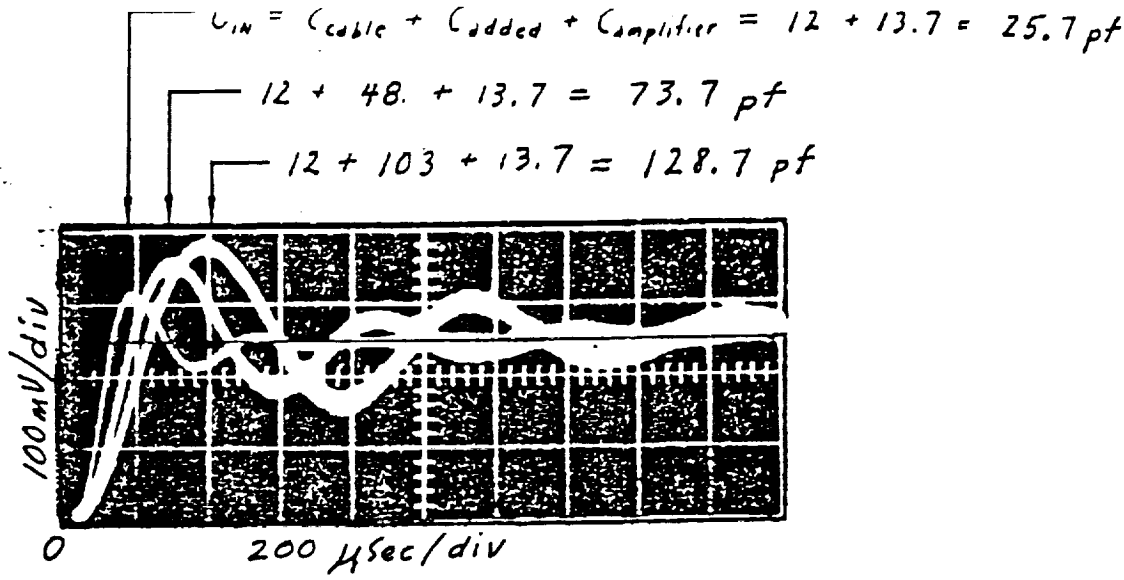


Figure.8a Step response for circuit in figure 2, with no compensation, and selected input capacitances.

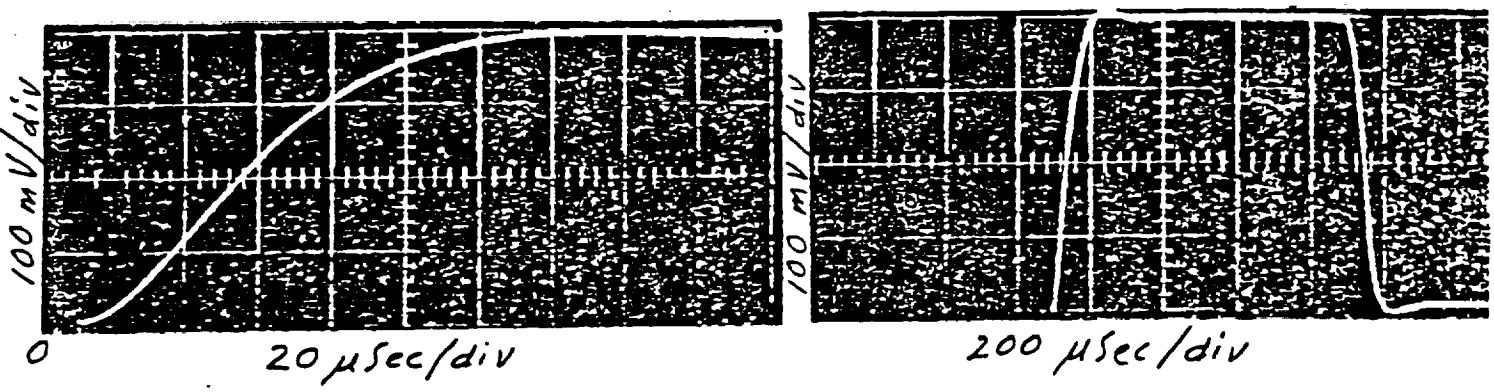


Figure 8b Typical compensated step response for circuit in figure 3, (second highest gain setting, C_{in} = 74.2 pf, R_{f4} = 10 M).

M E M O R A N D U M

February 13, 1986

MEMO TO: Advanced Langmuir Probes File (Project 021546)

FROM: B. P. Block

SUBJECT: Report on the Implementation and Performance of an Improved Langmuir Probe Amplifier

Introduction

The purpose of this study has been to improve the speed and accuracy of the Langmuir probe MK-2 pre-amplifier by use of a technique proposed by Dittmar and Macnee.[1] Their findings may be summarized as follows:

The MK-2 pre-amplifier was shown to be undercompensated for the effects of input and stray feedback resistor capacitances. When driven with a step of current, the circuit exhibited significant overshoot (as much as 90% when operated on the highest gain setting). Analysis showed that the amplifier, as configured, could never be adequately compensated for the range of expected input capacitances. Two new circuits were proposed. Each allowed compensation of input capacitance. In addition, the amplifier could be configured to produce a Thomson response characteristic[2], thus allowing a fast risetime with minimal overshoot.

Both circuits reduce the amount of signal fed to the range scaling feedback resistors by use of a voltage divider between

the operational amplifier output and ground. Because of this voltage reduction, the values of the range resistors must then be decreased in order to maintain the same amount of overall feedback. Thus, the effects of stray capacitances (made worse by large-valued feedback resistors) are greatly reduced.

One of the two new circuits allows the highest gain setting to be compensated with non-critical components, a significant improvement.

A penalty, which must be paid in this new approach, is an increase in pre-amplifier noise. The analysis predicts an increase of four times over the MK-2 figure. The noise characteristic is discussed in the section: Performance Measurements.

A laboratory model of the circuit described by Dittmar and Macnee has been constructed. The components and techniques that were employed are directly applicable to a flight program.

Implementation

A photograph of the finished laboratory pre-amplifier is shown in Figure 1. A stimulus box for providing current steps to the amplifer was also constructed and appears in the left side of the photograph.

Figures 2 and 3 are schematics of the circuit and laboratory model. A printed circuit board was constructed. The layout of this board is shown in Figure 4. One feature of the layout is the use of guard traces about the ranging resistors. These guards shield the feedback resistors from voltage gradients

generated by surface currents on the printed circuit board. This technique could enhance the amplifier's gain stability over long periods of time.

A specially-constructed brass box encloses the printed circuit board. This box is electrically insulated from and enclosed by an outer aluminum box. Power supply leads are carefully bypassed within the outer box to prevent noise from feeding through to the amplifier itself. The test signals are fed to the amplifier's input by triaxial cable. These precautions minimize the amount of externally induced noise and permit measurement of the amplifier's noise floor.

The amplifier was made to work with little trouble. Compensation capacitors were chosen to produce a minimum voltage overshoot at the output in response to a current step at the input in accordance with the Thomson time response characteristic. The highest gain range was easily compensated by a capacitance of 350 pF. This must certainly be considered a significant improvement over the old MK-2 circuit where capacitor values of a few picofarads were required.

Performance Measurements

As stated earlier, a significant increase in the amplifier's speed was desired. An improvement in measurement speed can be gained in the following ways:

1. Increasing the amplifier's risetime and reducing its overshoot (settling time).
2. Reducing the time required to recover from a step change in V_A.
3. Reducing the time required to recover from a range change.

The amplifier's performance was studied in all of these areas.

The amplifier was configured for four gain ranges. Input current is converted to output voltage, therefore the amplifier gain is specified by its transresistance in ohms. The approximate gains are shown below:

Range	Transresistance
1	1000 Megohms
2	75 Megohms
3	4 Megohms
4	225 Kilohms

Table 1

Oscilloscope photographs of the current-step response of the amplifier appear in Figure 5. Only the first three ranges are shown; the risetime of the fourth was much faster than the first three and would not be a factor in total measurement time. In fact, it can be seen that the risetime of Range 1, the highest gain setting, is by far the slowest of all the ranges.

A modified measurement algorithm has been proposed and was investigated in an earlier study[3]. A key requirement for the success of this algorithm is the ablility of the pre-amplifier to

rapidly recover from a voltage step in V_A. These measurements are shown in Figure 6. Again, it can be seen that the highest gain range requires the greatest time to recover. In this case, the settling time is on the order of 0.5 mS. A further factor which influences recovery time is the rate of change of V_A. The test signal used changes at the rate of 0.5 V/uS, a reasonable value for a D/A converter. The settling time for the amplifier appeared to be independent of the direction of the V_A step.

A third, but no less critical, requirement is the ability to recover from a range change. These measurements are shown in Figure 7. The longest recovery time observed occurred when the gain was changed from the 1000 M range to 75 M range. This downrange required nearly 0.8 mS. It can be seen that the amplifier saturates and then recovers. Dittmar and Macnee reported that step response times could be improved significantly by using an operational amplifier with a higher gain-bandwidth product. Although recovery from saturation is not necessarily related to gain-bandwidth product, an HA2510 operational amplifier was substituted into the circuit. This amplifier's GB product is about ten times that of the HA2700. The recovery time improved significantly.

The amplifier's noise floor was measured on all ranges and was found to be largest on the 1000 M range. The noise appeared to be wideband with a root-mean-square amplitude of 5 mV which corresponds to an input current of 5×10^{-12} amperes, a value that is probably small enough to be of no concern.

Conclusions

Dittmar and Macnee showed that the MK-2 amplifier possessed significant overshoot and ringing on all ranges. (Table 1 of Ref. 1) The worst case settling time was about 2 mS for a current-step input. The new version, as just discussed, improves this figure by nearly an order of magnitude.

Unfortunately, measurements were not made on the recovery times of the original MK-2 for changes in V_A and gain range. It is reasonable to expect, however, that the recovery from steps in V_A has been improved significantly, most likely by about the same factor. The case of recovery from range changes is not quite as clear-cut. Certainly the MK-2 amplifier's output must have saturated when ranging occurred. Most likely, recovery from saturation was similar to that seen in Fig. 7, but the settling time would certainly have been much longer. Therefore, one may conclude that the response of the circuit has been improved in each of the three areas outlined above.

It has also been demonstrated that a faster operational amplifier can significantly improve the recovery time from a range change. This recovery time seems to determine the minimum period in which a measurement could be made. Selection of a higher-speed device for a Langmuir probe pre-amplifier would be done within the constraints of a particular flight program.

The exact determination of this measurement period, as well as a demonstration of the modified measurement algorithm, will be the goal of the next phase of this study.

References

1. Dittmar, J. A. and Macnee, A.B.,"An Analytic and Experimental Study of Variable Gain Langmuir Probe Circuits", internal report dated July 28, 1983.

2. Blinchikoff, H.J. and Zverev, A.I., Filtering in the Time and Frequency Domain, John Wiley and Sons, New York, 1976, pp. 124-127.

3. Miller, W. A. "Ann Arbor Advanced Adaptive Probe Algorithm", a computer program in FORTRAN IV, May 1, 1985.

An Improved Langmuir Probe Amplifier

List of Figures

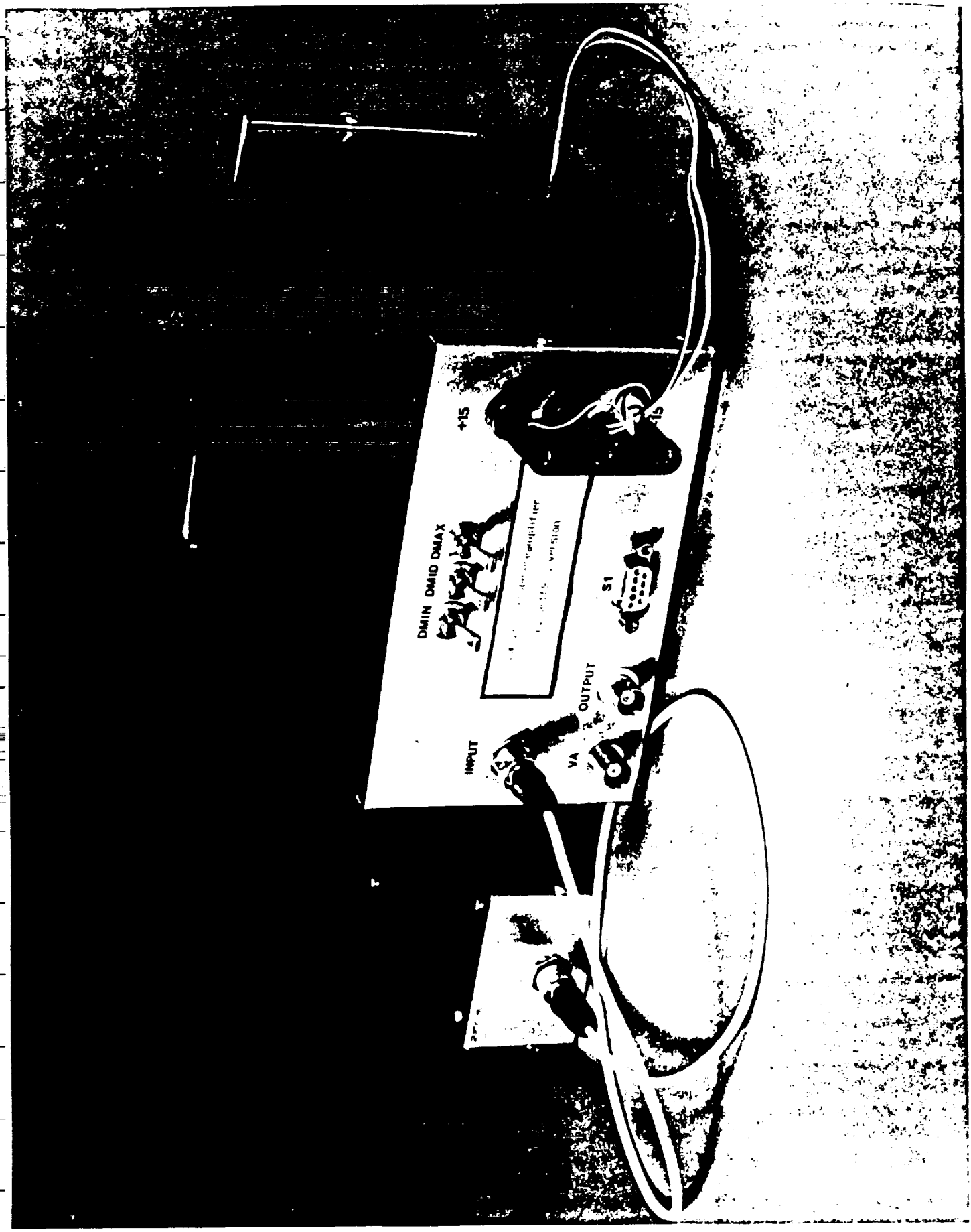


Figure 1

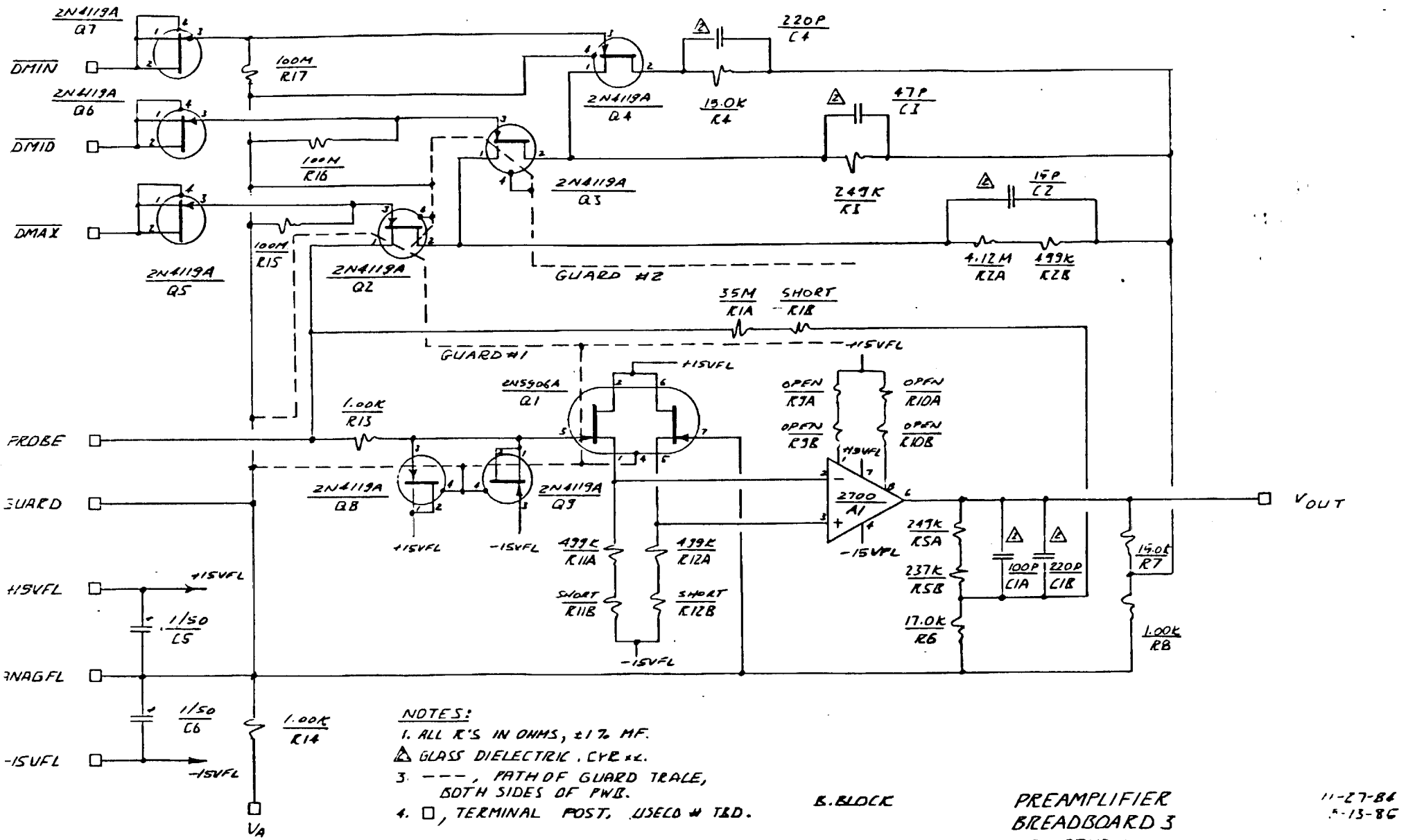
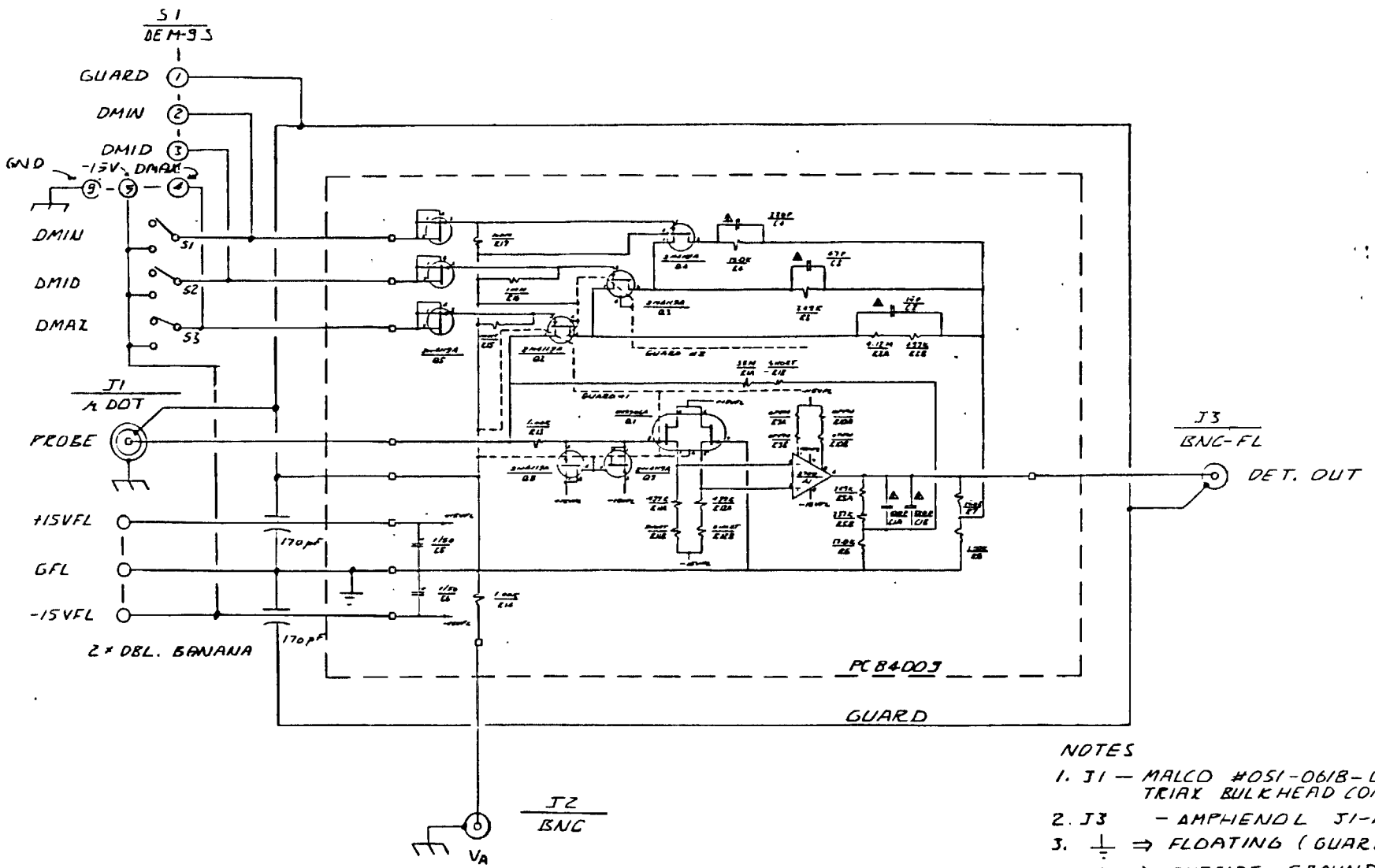

2N4119A Q7
DMIN
100M R17
2N4119A Q6
DMID
100M R16
2N4119A Q5
DMAX
100M R15
2N4119A Q2
2N4119A Q4
220P C4
15.0K R4
2N4119A Q3
47P C3
249K R3
15P C2
4.12M R2A
499K R2B
GUARD #2
35M R1A
SHORT R1B
GUARD #1
+15VFL
2N5906A Q1
1.00K R13
PROBE
GUARD
2N4119A Q8
2N4119A Q9
OPEN R9A
OPEN R9B
OPEN R10A
OPEN R10B
2700 A1
-15VFL
499K R11A
SHORT R11B
499K R12A
SHORT R12B
249K R5A
237K R5B
17.0K R6
100P C1A
220P C1B
15.0K R7
1.00K R8
VOUT
+15VFL
1/50 C5
-15VFL
1/50 C6
1.00K R14
VA
NOTES:
1. ALL R'S IN OHMS, ±1% MF.
2. GLASS DIELECTRIC. CVR ±2.
3. ---, PATH OF GUARD TRACE, BOTH SIDES OF PWB.
4. □, TERMINAL POST, USECO W T&D.
B. BLOCK
PREAMPLIFIER BREADBOARD 3 LP STUDY
11-27-84
5-13-85

Figure 2



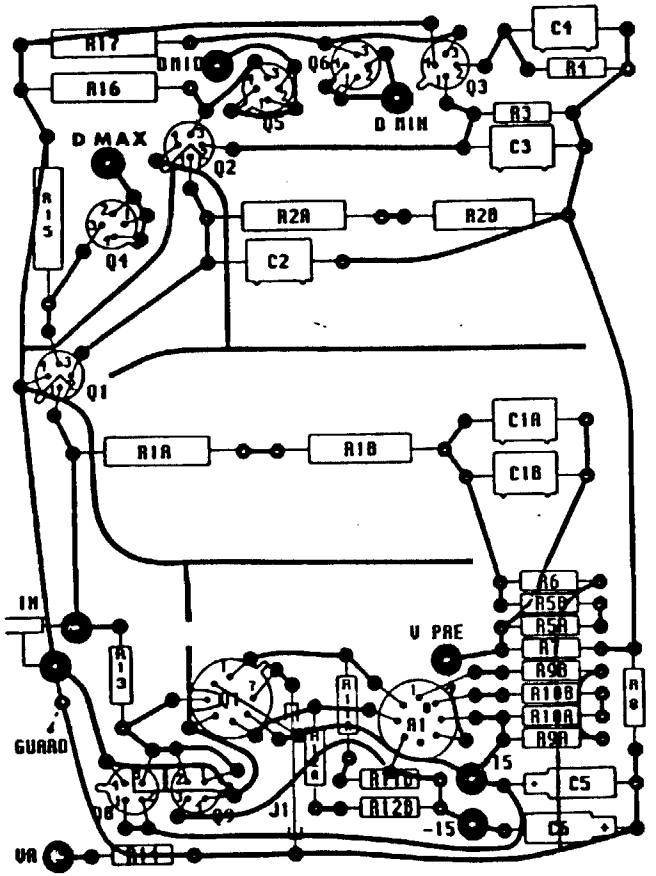
NOTES

1. J1 — MALCO #051-0618-0001 TRIAX BULKHEAD CONNECTOR
2. J3 — AMPHENOL 51-10
3. ⏚ ⇒ FLOATING (GUARD) GROUND
   ⏚ ⇒ OUTSIDE GROUND

B. BLOCK . PREAMPLIFIER TEST BOX 7-11-85
BREADBOARD 3 8-13-85
LF STUDY

Figure 3

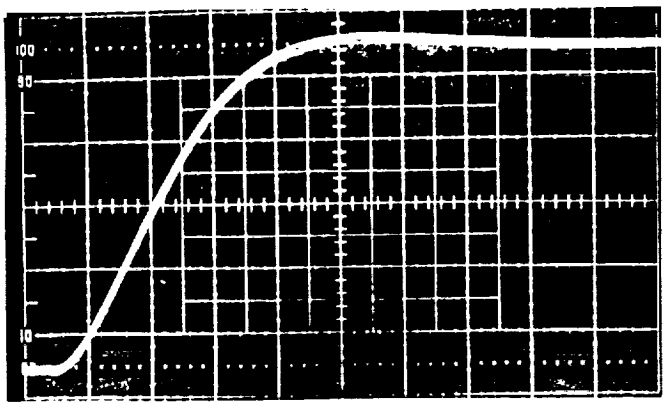
COMPONENT SIDE



COMPONENT SIDE CIRCUIT SIDE

ENGINEER B. BLOCK		DATE
DRAFTER	8/13/85	
PC PICTORIAL BB #3 ADV. LANGMUIR PROBE		
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN		
B-E7729		

Figure 4



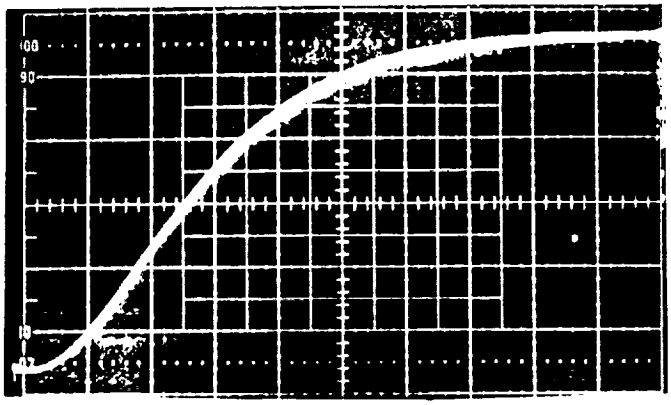
a.

Z_t = 1000 M

Vert: 1 V/div

Horiz: 100 uS/div

t_r = 250 uS



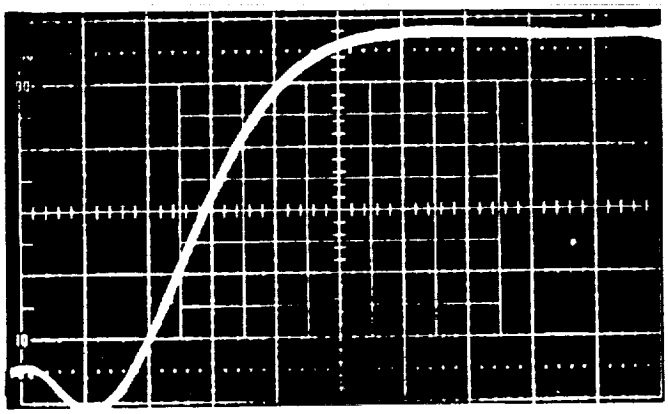
b.

Z_t = 75 M

Vert: 1 V/div

Horiz: 20 uS/div

t_r = 80 uS



c.

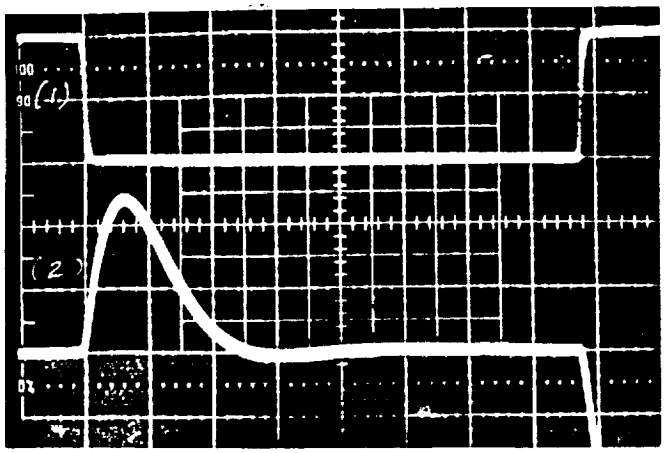
Z_t = 4 M

Vert: 0.1 V/div

Horiz 10 uS/div

t_r = 25 uS

Figure 5 Response to a Current Step



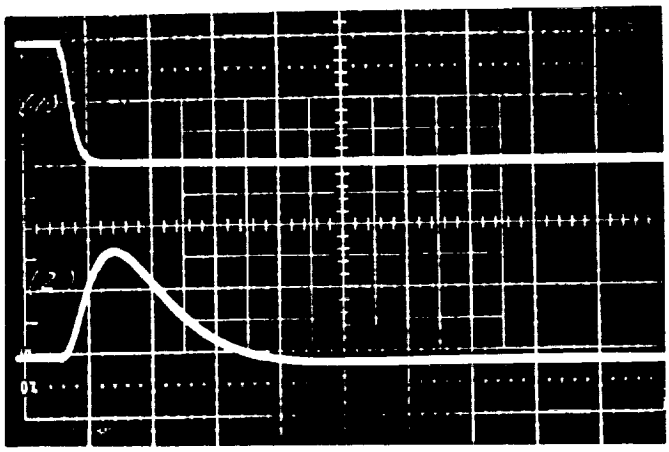
a.

Z_t = 1000 M

Vert: (1) V_A , 5 V/div
(2) V_{out} ,0.1 V/div

Horiz: 200 uS/div

t_s = 550 uS



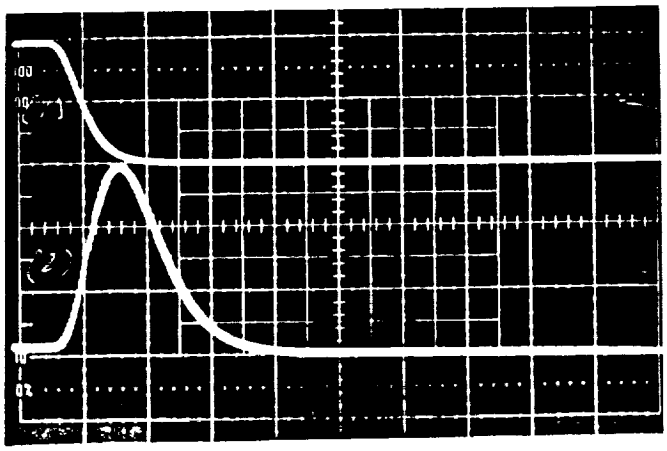
b.

Z_t = 75 M

Vert: (1) V_A , 5 V/div
(2) V_{out} , 0.1 V/div

Horiz: 50 uS/div

t_s = 150 uS



c.

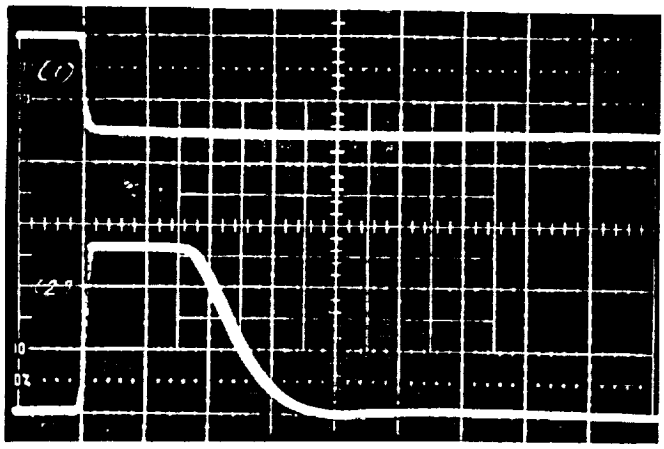
Z_t = 4 M

Vert: (1) V_A , 5 V/div
(2) V_{out} ,50 mV/div

Horiz 20 uS/div

t_s = 75 uS

Figure 6 Response to a Step in V_A



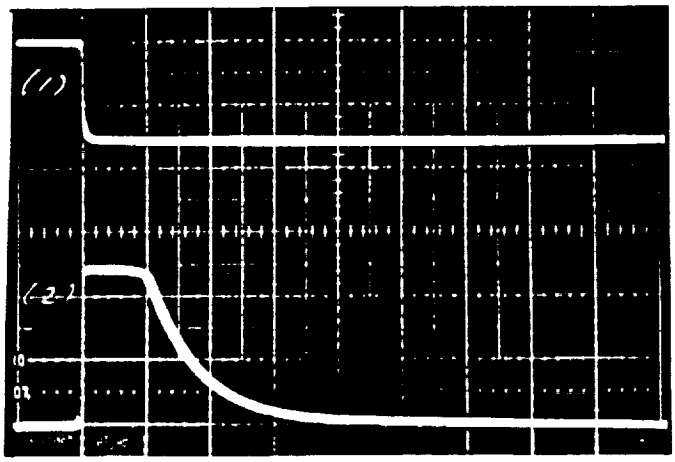
a.

HA2700

Vert: (1) V_{SW}, 10 V/div
(2) V_{out}, 5 V/div

Horiz: 200 uS/div

$t_{recovery}$ = 800 uS



b.

HA2510

Vert: (1) V_{SW}, 10 V/div
(2) V_{out}, 5 V/div

Horiz: 100 uS/div

$t_{recovery}$ = 400 uS

Figure 7 Recovery from a Range Change -- Downrange from Z_t = 1000 M to 75 M

M E M O R A N D U M

87/008

June 4, 1987

MEMO TO: Advanced Langmuir Probe File (Project 021546)

FROM: B. P. Block

SUBJECT: Report on the Implementation of a Dedicated Processor for the Advanced Langmuir Probe

Introduction

The auto-framing algorithm used in the Pioneer Venus and DE-2 instruments was implemented with hardwired, discrete logic. The purpose of this study has been to design and fabricate a replacement for this hardwired logic using a programmable controller and to demonstrate a new auto-framing algorithm programmed in the controller's software.

Over the past few years, the dramatic advances in microcomputing and low-power integrated circuit technology have made practical the incorporation of programmable devices in spaceflight instruments. Control functions formerly encoded in discrete hardware can now be represented as instructions to a programmable controller (usually a microprocessor); through associated hardware, the controller can cause the sensor to make a measurement and then report the results. Principal among the advantages of this approach is the ability to employ measurement algorithms of much greater complexity than those that were feasible with discrete logic.

A new measurement technique, the five-point algorithm (described in a previous report), takes advantage of the computational capabilities of the controller to measure discrete values of the volt-ampere (V-A) function. This alogorithm rapidly frames the V-A curve thereby allowing high temporal resolution as well as immediate recovery from fault conditions. It is hoped that an attendant reduction in data rate will also follow.

The instrument measurement and control algorithms are described by software contained in the controller's program memory and are given life by the central processor unit as the program executes. Unlike general computing software, the instrument software must exhibit a time-responsiveness: the

sensor must make measurements at precise intervals, the results must be telemetered periodically, and commands must be received asynchronously. Hardware and software exhibiting this time-responsiveness constitute a real-time system.

Real-time Systems

A real-time system is a collection of devices controlled and monitored by a stored program of instructions. Stimuli from these devices, in the form of interrupts, initiate and influence program processing; the software is said to be interrupt-driven. Although the device stimuli arrive at unpredictable intervals, the software responses must nevertheless be delivered within rigid time constraints. For example, commands arriving from the spacecraft must be given immediate attention, even though the instrument software is busy controlling a measurement cycle. In reaction to these commands, the software must respond correctly and completely by altering the instrument state to accomplish the desired operation. It must also do so reliably, perhaps even in the event of hardware failure.

These requirements have presented a special challange in the design of the programmable controller. The hardware and software solutions to the problems of responsiveness, correctness and reliability will now be discussed.

Hardware

Many programmable devices exist in the marketplace today. A few of the requirements for a programmable Langmuir probe controller are:

1. Low power consumption
2. Demonstrated reliability, qualifiable to the relevant program standards
3. An architecture suitable for use with high-level languages, such as FORTRAN or C
4. The availability of hardware/software development tools.

A survey of existing devices was undertaken. Only one device, the Harris 80C86 microprocessor, met all of the above requirements. This device is a general-purpose 16-bit CMOS microprocessor fabricated using a self-aligned silicon gate CMOS process. The 80C86 is capable of addressing a maximum of

1 megabyte of program memory and 64 kilobytes of input/output (I/O). The 80C86 requires a system clock rate of 5 MHz and typically consumes 250 milliwatts of power. The device has been qualified by NASA for a number of flight programs, including the UARS/High Resolution Doppler Imager. Many development tools exist at SPRL: a cross-assembler, several high-level language cross-compilers, and an in-circuit emulator. These tools are discussed in the Software section of this report.

The concept of the controller, hereafter called the Dedicated Processor (DP), is shown in Figure 1. The 80C86 CPU and associated clock generator are shown on the left-hand side of the drawing. Program memory consists of 32 kilobytes of erasable programmable read-only memory (EPROM) and 20 kbytes of static random access memory (SRAM). The system is capable of responding to a maximum of 8 external interrupts by use of a programmable interrupt controller (PIC). To provide system event timing, a three-channel programmable interval timer (PIT) was incorporated. This device interrupts the processor at a 100 Hz rate, initiating measurement cycles, scheduling system tasks, etc. Communication with the DP, in lieu of a specific command and telemetry interface, is accomplished by a standard RS232 computer connection. This is a convenient, low-cost method of acquiring data and sending commands. The inclusion of the more usual PCM telemetry and command interface for use in a flight instument would be straightforward. Because the full one megabyte memory space is not used, a means of detecting accesses to unused addresses was added. Such an access produces a warning message through the RS232 serial port.

The hardware controlled by the DP is called the signal processor and is shown in Figure 2. This hardware, developed during an earlier study (NAS5-26678), produces the voltage applied to the probe and measures the resulting probe current. Communication between the DP and signal processor occurs through a 16-bit bi-directional digital interface. The signal processor appears to the DP as a series of registers through which data can be passed. Additionally, the signal processor interrupts the DP at the end of every probe current conversion.

The exact way in which memory and I/O are arranged is shown in Figure 3.

Thus, the DP, signal processor, and instrument power supply (not shown) constitute the Langmuir probe electronics. Appendix A contains a complete schematic set and wire listing for the DP.

The dedicated processor requires a 5 volt power supply and consumes approximately 0.9 watts of power.

The careful selection of components throughout the DP makes the transport of this design to a flight-qualifiable version relatively straightforward. Certain features that may enhance the reliabilty of a flight system were not incorporated. Any flight unit would likely include a watchdog timer and redundant memory. However, these features could be readily added to the current design.

Software

Like the hardware that it replaces, the DP software is substantially complex. Many software instructions replace individual logic gates. Given the control and measurement algorithms to be implemented, the potential complexity of the software is great. A way to reduce this complexity and its cost is to impose a strong modular structure on the software design. If the software is not constructed in a highly structured manner, testing becomes difficult, if not impossible. The stringent reliablity requirements imposed on a flight version of the Advanced Langmuir Probe system would require that each software module be thoroughly tested before it is incorporated into the DP.

In order to enforce the necessary structure at the system level, a well-tested, commercial software package called the Ready Systems Virtual Real-time Executive (VRTX) is used. VRTX has been approved by NASA-GSFC for use on the UARS program and is presently being evaluated by JPL for use on the Mariner Mark II spacecraft.

The features contained in VRTX include the following:

1. Multitasking support - a number of tasks can be handled concurrently.

2. Interrupt-driven, priority-based scheduling - tasks are assigned a priority level according to their time-criticality.

3. Intertask communication and synchronization - tasks can send messages through mailboxes and queues; semaphores can be employed for synchronizing two or more tasks.

4. Dynamic memory allocation - a task can request the use of a certain amount of memory and, when finished, relinquish it.

5. Real-time clock control - timing is referenced to a hardware clock.

6. Character I/O support - a standard means of communicating with one or more terminals is provided.

7. Real-time responsiveness - the system responds to outside events rapidly enough to control the ongoing process.

By dividing the software into a number of independent, well-defined tasks, each of which can be specified, written and tested individually, the desired modularity can be obtained. VRTX manages each of these tasks concurrently, providing standard services, such as those listed above.

Under VRTX, the measurement algorithm, written as one or more tasks, is scheduled, executed and then suspended until the next measurement cycle. In the meantime, other tasks such as the telemetry task are scheduled and executed. Communication occurs regularly between tasks and the system terminal.

Because the principal cost of a microprocessor-based system lies in its software, not in the hardware, the easy portability of the software from one type of microprocessor to another is a highly desirable goal. This is especially true for the Advanced Langmuir Probe, where no particular flight program has been identified and the program-specific part requirements have not been established. VRTX, available for many different microprocessors, will allow those routines coded in a high- level language to be transported directly. Only the board-support routines, written in native code, would need to be altered.

The system structure of the Advanced Langmuir Probe is shown schematically in Figure 4.

The SPRL-written software consists of board-level support routines necessary for VRTX and the application code itself. The application code implements the measurement algorithm, the terminal communications and all I/O to the signal processor. This software will be written in the C programming language.

At present, VRTX has been installed on the DP and a number of simple tasks have been written to verify that it functions properly.

Certain development tools are necessary for the writing and debugging of DP software. The Space Physics Research Laboratory has an extensive collection of cross-development software for the 80C86 microprocessor. The schema for DP software development is shown in Figure 5.

The DP, having only a small amount of memory and no secondary storage devices, is not a viable machine for developing software. Consequently, a much larger computer, a DEC VAX 8600 mainframe, is used for generating the object code to be executed by the DP. A number of cross-assemblers and cross-compilers reside on the VAX 8600. The product of these is an executable image that is downloaded to a mini-computer. This minicomputer, a Charles River Data System LSI-11/23, stores the image locally and then transmits the code to the DP. Data acquired from the DP/signal processor can be stored on the disk of the LSI-11/23 and processed for presentation in graphical or tabular form. In addition to VRTX and the application code, a small resident monitor, MON86, and a software package called TRACER reside on the DP. TRACER is designed to interactively monitor the execution of VRTX by displaying task status on the user console. Additionally, it communicates with the LSI-11/23 to accomplish the downloading of the object image.

A listing of the board-support software for VRTX and TRACER as well as the source code for MON86 is contained in Appendix B.

Present Status

The installation and testing of VRTX has been completed. Some simple programs (tasks) have been developed to verify that both VRTX and the DP hardware operate reliably. The ability to transfer object files between the VAX 8600 and the CRDS LSI-11/23 has been demonstrated.

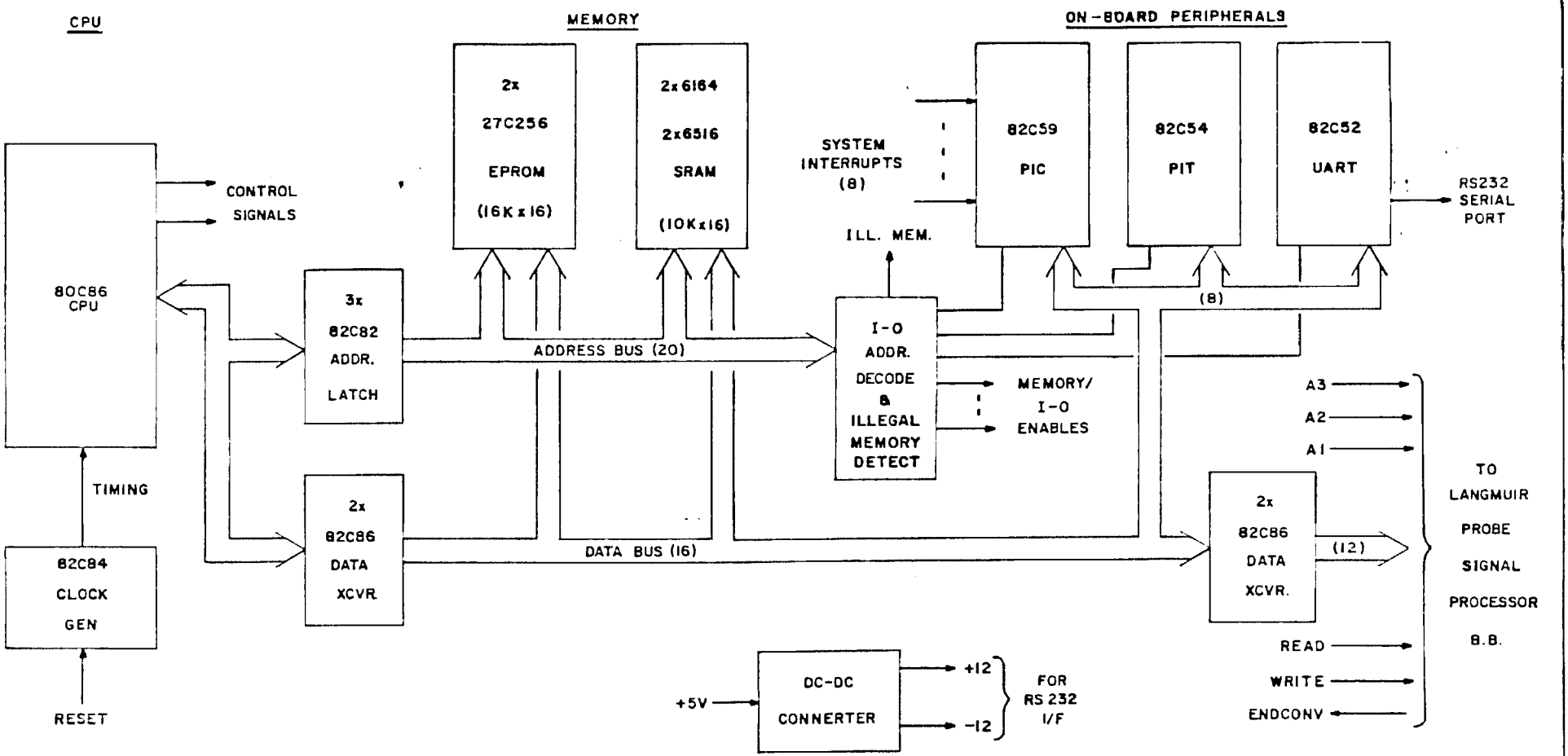
Work is currently centered on the specification of the ALP application software and the design and coding of this software in the C programming language. Development of software for the graphical display of the acquired data is being developed in parallel. The ability to display, in graphical form, the V-A funtion on a high-resolution CRT, as well as on hardcopy, will allow easy evaluation of the five-point algorithm and the fidelity of the plasma simulators.

Conclusion

A low-power, reliable, microprocessor-based controller has been designed, built and tested. A real-time executive has been installed and tested on this controller. This combination of hardware and software has yielded a versatile, general-purpose, programmable replacement for the hardwired logic of the Langumuir probe electronics. Its utility is not limited to this particular instrument; with little or no modification, only the installation of the appropriate application software, it could be used to control a wide variety of scientific instruments.

Work continues on the application software necessary to implement the five-point algorithm and the graphics display. With this modest continued effort, a demonstration of a functioning Advanced Langmuir Probe will be possible in the near future.

List of Figures



ENGINEER & BLOCK	DRAFTSMAN R.
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR MICHIGAN	CONCEPT DWG. DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE B-E 7926
	9/18/85 DATE

LAST USED R C D L

1.0

FIGURE 1

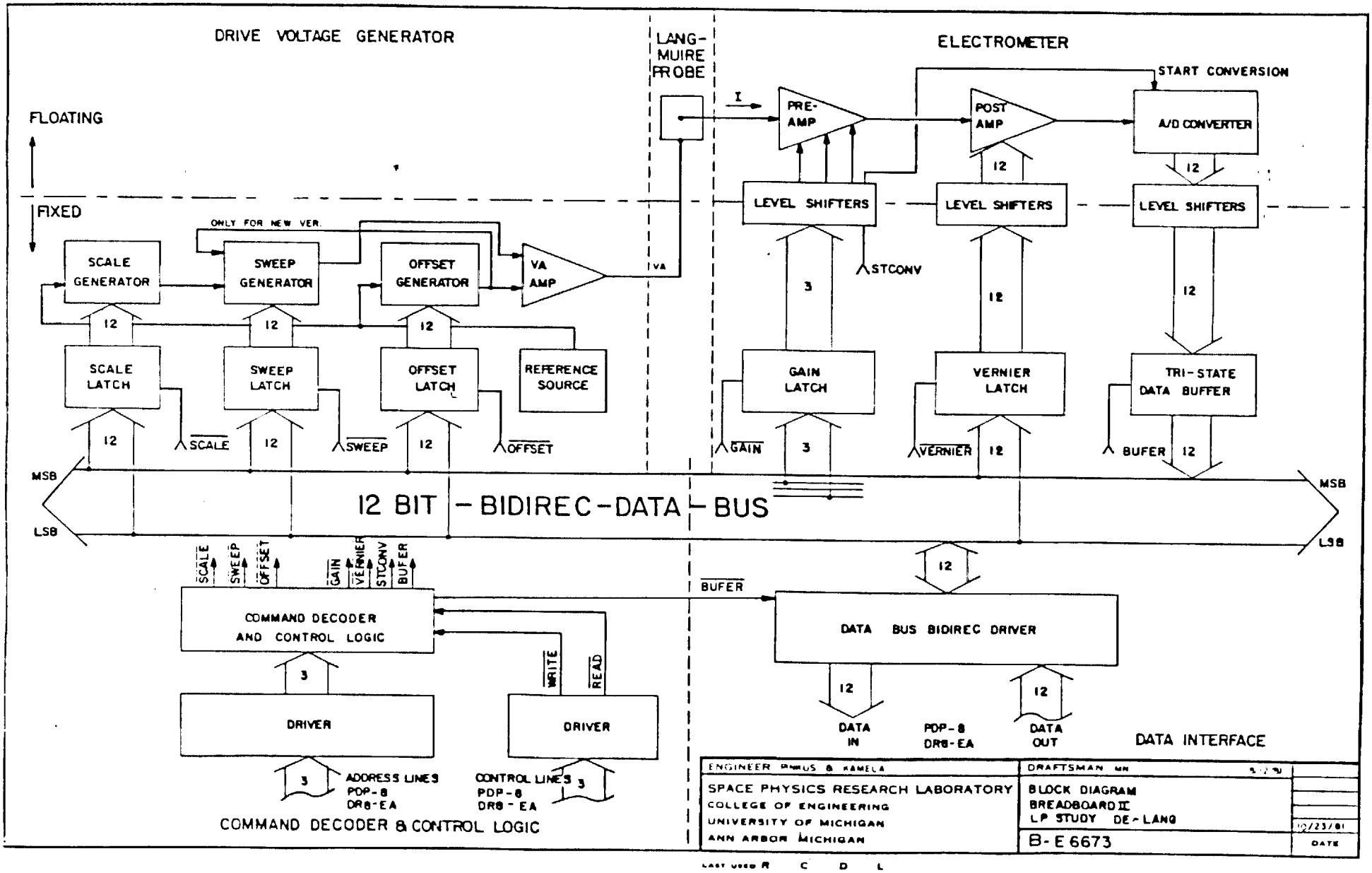
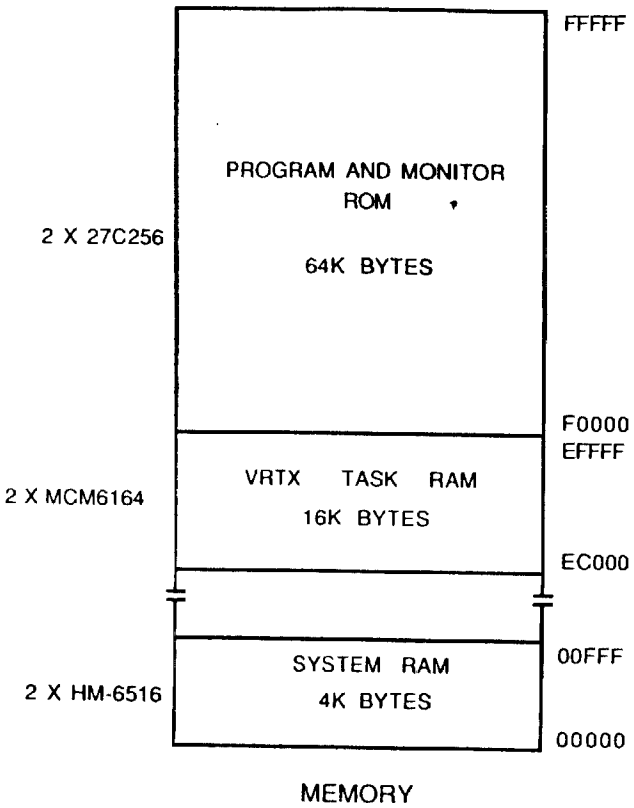
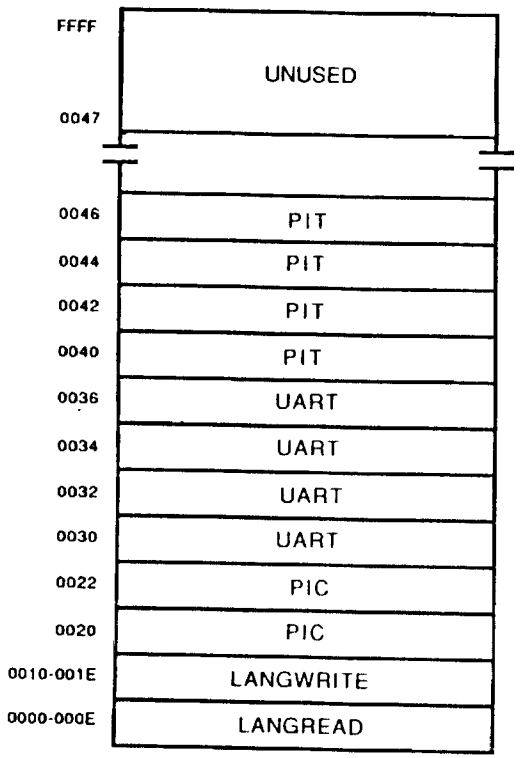

DRIVE VOLTAGE GENERATOR
LANG-MUIRE PROBE
ELECTROMETER
FLOATING
FIXED
START CONVERSION
I
PRE-AMP
POST AMP
A/D CONVERTER
ONLY FOR NEW VER.
SCALE GENERATOR
SWEEP GENERATOR
OFFSET GENERATOR
VA AMP
VA
LEVEL SHIFTERS
LEVEL SHIFTERS
LEVEL SHIFTERS
STCONV
SCALE LATCH
SWEEP LATCH
OFFSET LATCH
REFERENCE SOURCE
GAIN LATCH
VERNIER LATCH
TRI-STATE DATA BUFFER
SCALE
SWEEP
OFFSET
GAIN
VERNIER
BUFER
MSB
LSB
12 BIT - BIDIREC - DATA - BUS
STCONV
BUFER
COMMAND DECODER AND CONTROL LOGIC
DATA BUS BIDIREC DRIVER
WRITE
READ
DRIVER
DRIVER
ADDRESS LINES PDP-8 DR8-EA
CONTROL LINES PDP-8 DR8-EA
DATA IN
PDP-8 DR8-EA
DATA OUT
DATA INTERFACE
COMMAND DECODER & CONTROL LOGIC
SPACE PHYSICS RESEARCH LABORATORY
COLLEGE OF ENGINEERING
UNIVERSITY OF MICHIGAN
ANN ARBOR MICHIGAN
BLOCK DIAGRAM
BREADBOARD II
LP STUDY DE-LANG
B-E 6673
10/23/81

FIGURE 2



MEMORY



I/O

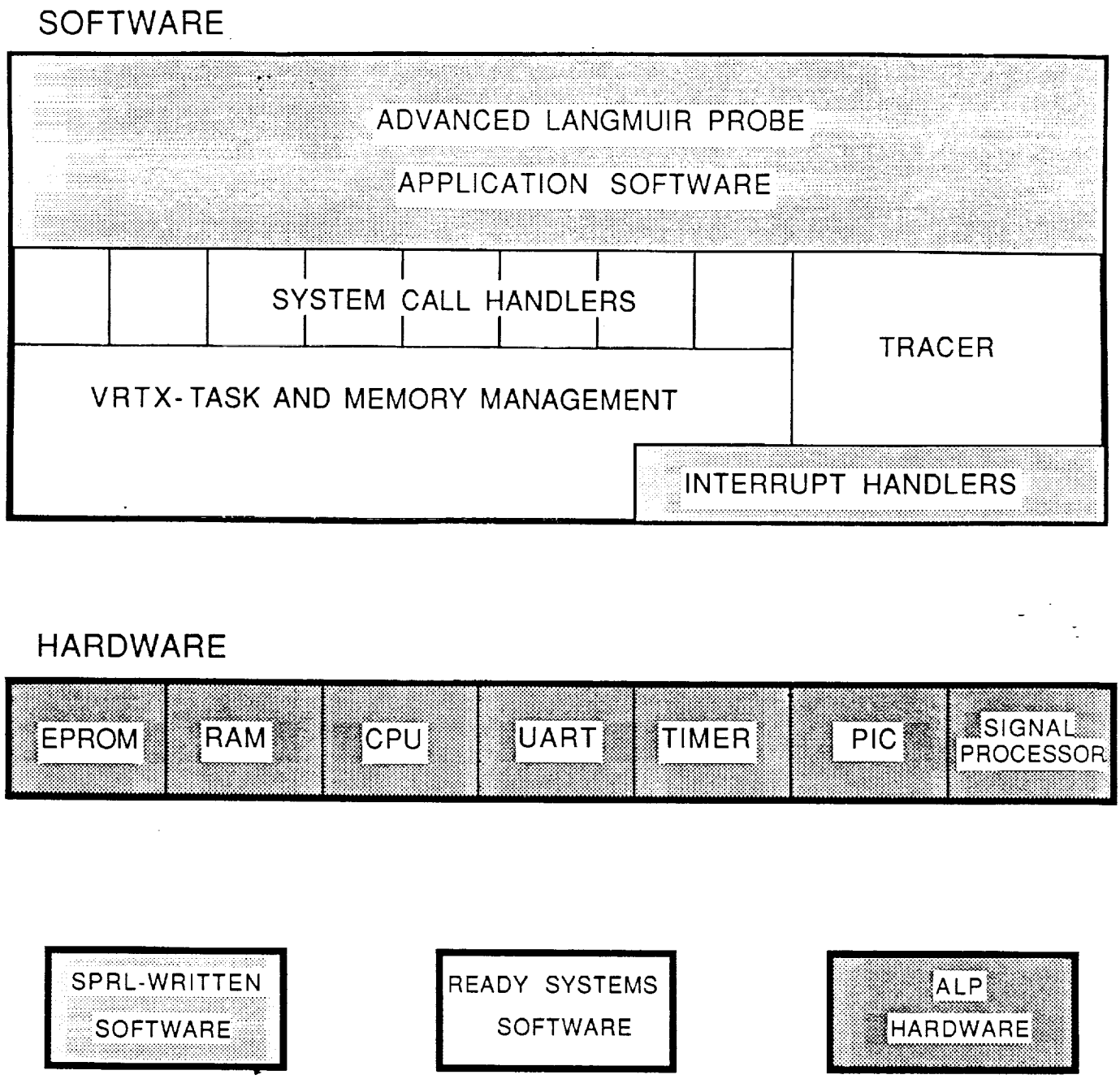
NOTE:

ADDRESSES ARE IN HEXIDECIMAL

MEMORY AND I/O MAP
DEDICATED PROCESSOR
ADVANCED LANGMUIR PROBE

5/20/87 BPB

FIGURE 3

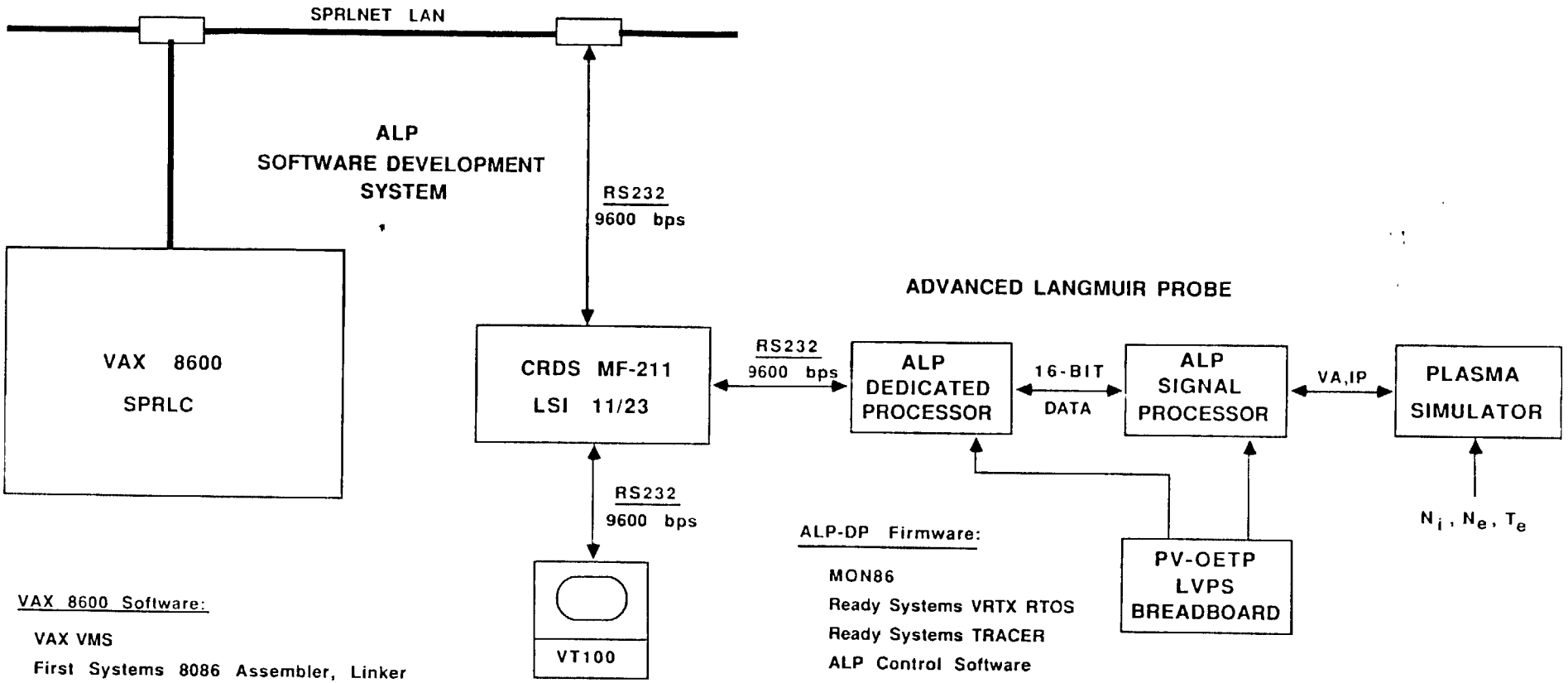


SYSTEM STRUCTURE

ADVANCED LANGMUIR PROBE

5/22/87 BPB

FIGURE 4



SOFTWARE DEVELOPMENT SCHEME
DEDICATED PROCESSOR
ADVANCED LANGMUIR PROBE

FIGURE 5

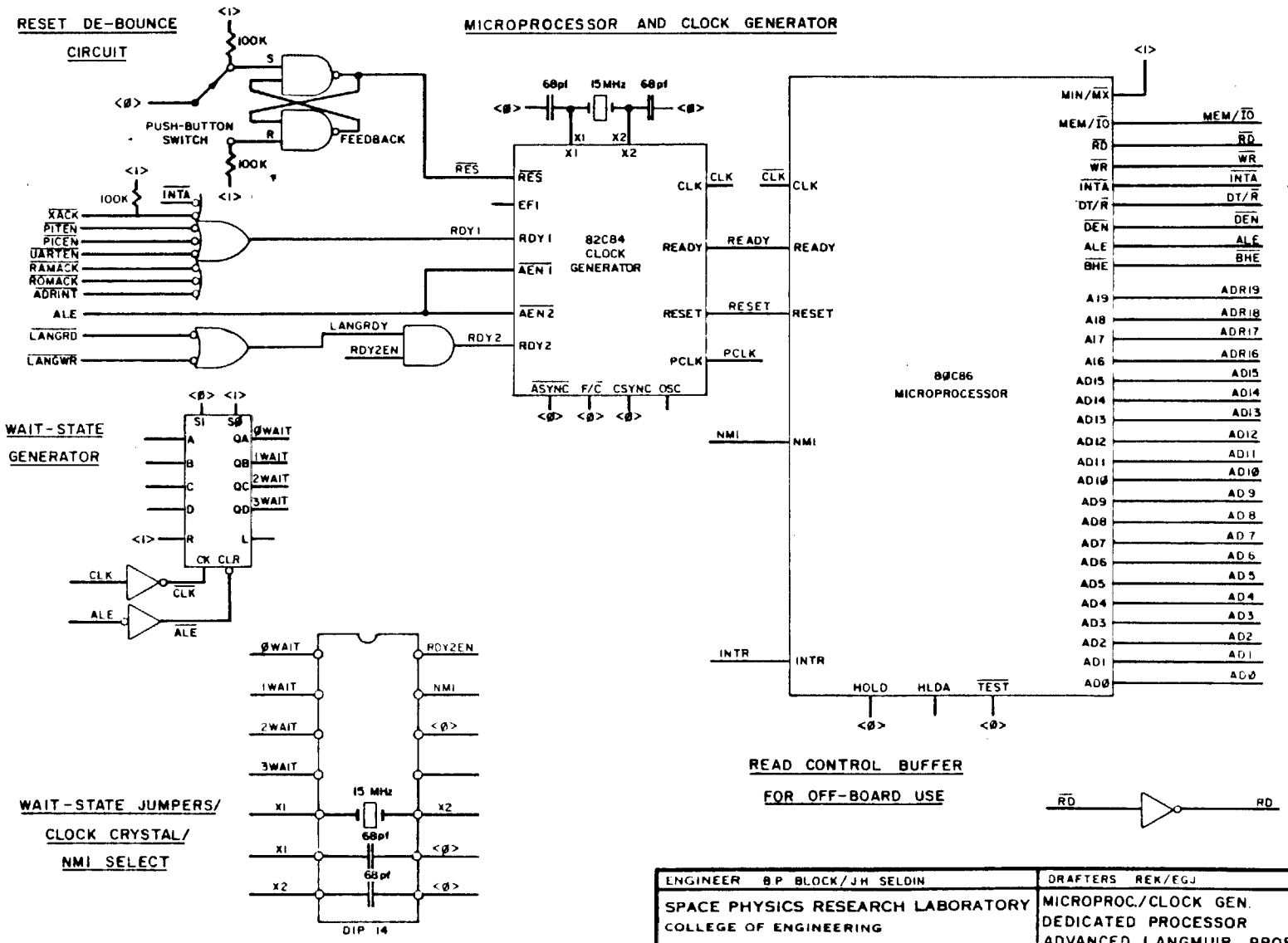
5/20/87 BPB

Appendix A

Schematic Set and Wirelists

for the

Dedicated Processor

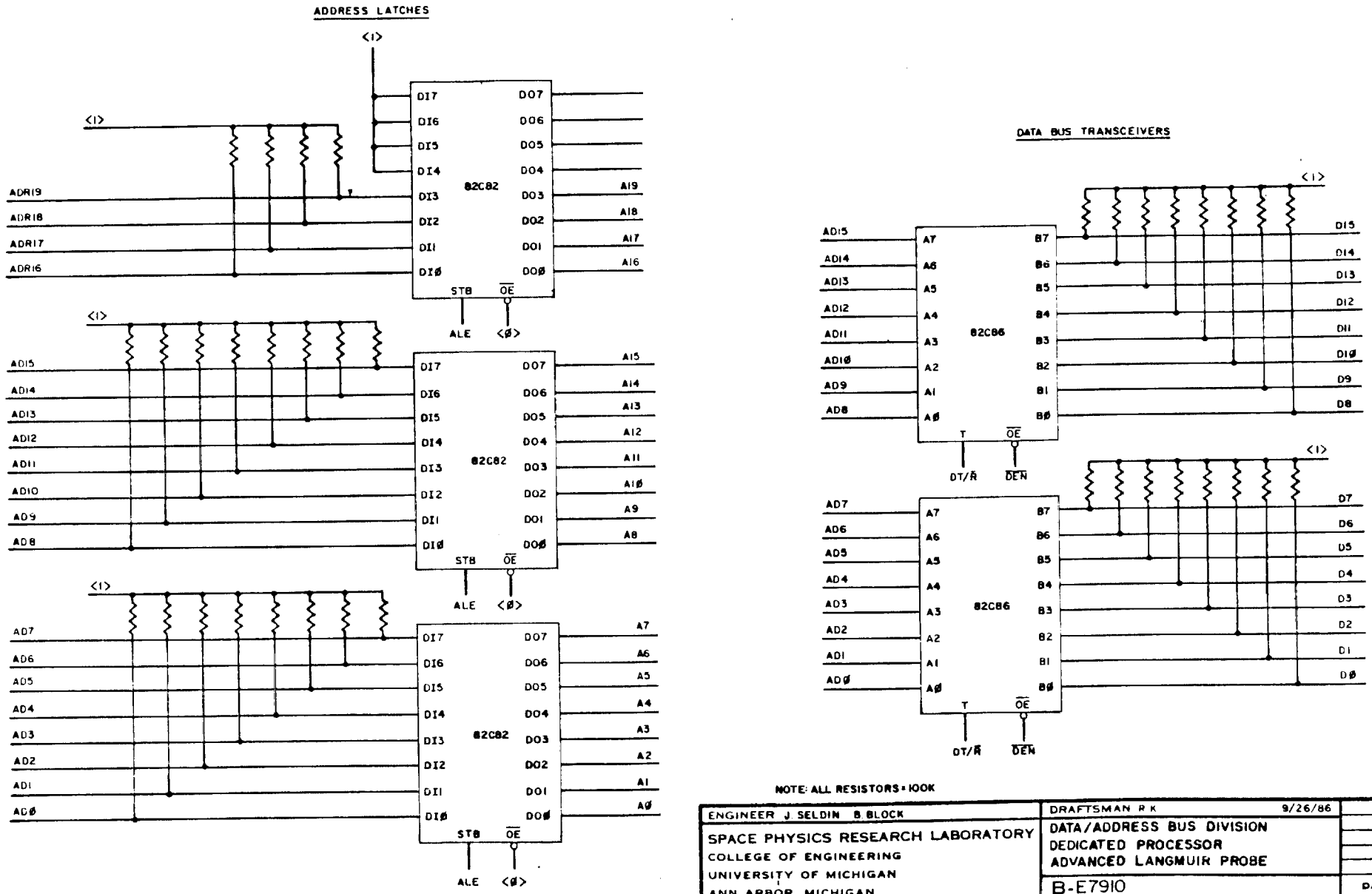


ENGINEER BP BLOCK/JH SELDIN	DRAFTERS REK/EGJ 9/18/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	MICROPROC./CLOCK GEN. DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	6/3/87
	B-E7902	DATE

K&E 10 1153 2/86 1140

LAST USED R C D L

1.1

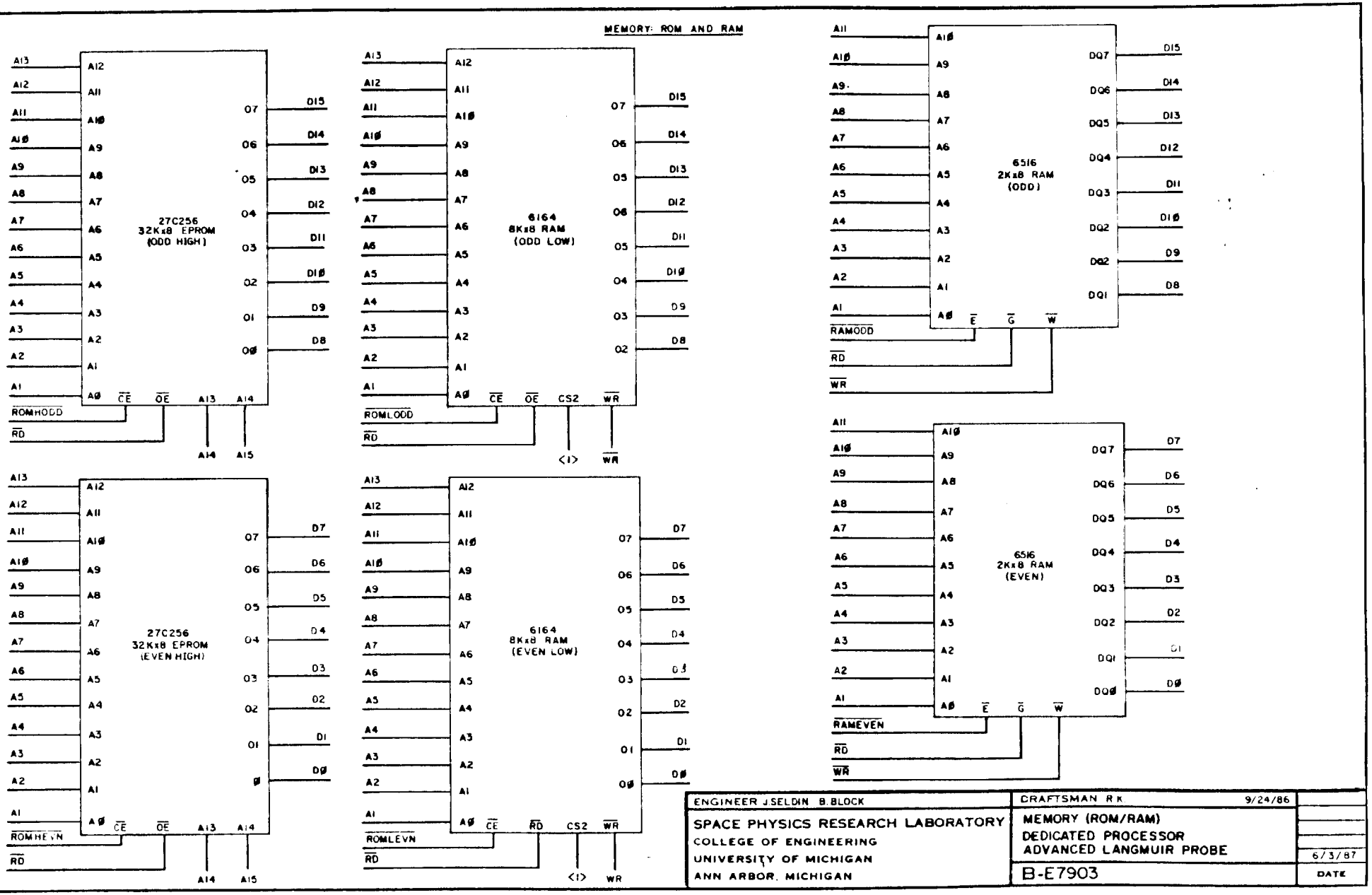


NOTE: ALL RESISTORS = 100K

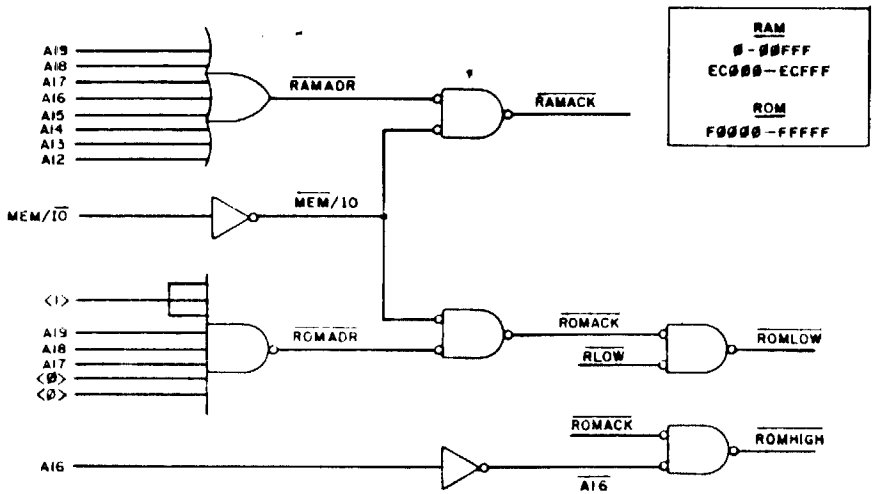
ENGINEER J. SELDIN B. BLOCK	DRAFTSMAN R K 9/26/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	DATA/ADDRESS BUS DIVISION DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	
	B-E7910	DATE

K&E 10 1153 2/86 1140F

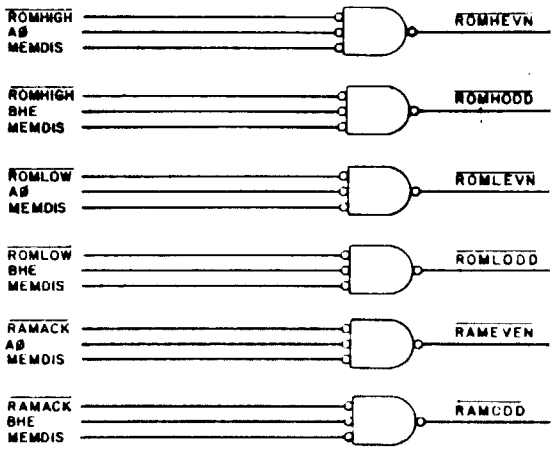
LAST USED R C D L

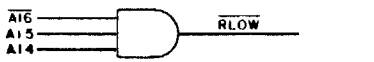

MEMORY: ROM AND RAM
27C256
32Kx8 EPROM
(ODD HIGH)
6164
8Kx8 RAM
(ODD LOW)
6516
2Kx8 RAM
(ODD)
27C256
32Kx8 EPROM
(EVEN HIGH)
6164
8Kx8 RAM
(EVEN LOW)
6516
2Kx8 RAM
(EVEN)
ROMHODD
ROMLODD
RAMODD
ROMHEVN
ROMLEVN
RAMEVEN
RD
WR
CE
OE
CS2
ENGINEER J.SELDIN B.BLOCK
CRAFTSMAN R.K. 9/24/86
SPACE PHYSICS RESEARCH LABORATORY
COLLEGE OF ENGINEERING
UNIVERSITY OF MICHIGAN
ANN ARBOR, MICHIGAN
MEMORY (ROM/RAM)
DEDICATED PROCESSOR
ADVANCED LANGMUIR PROBE
B-E7903
6/3/87
DATE
LAST USED R C D L

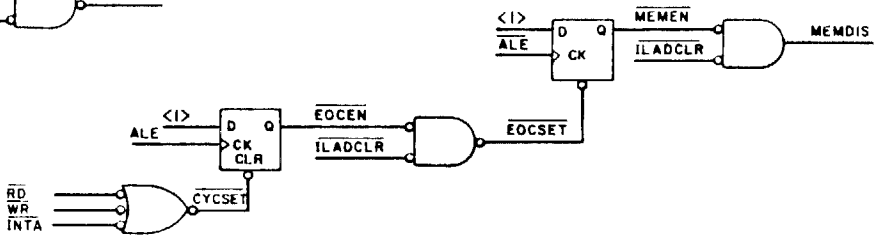
MEMORY ADDRESS DECODE



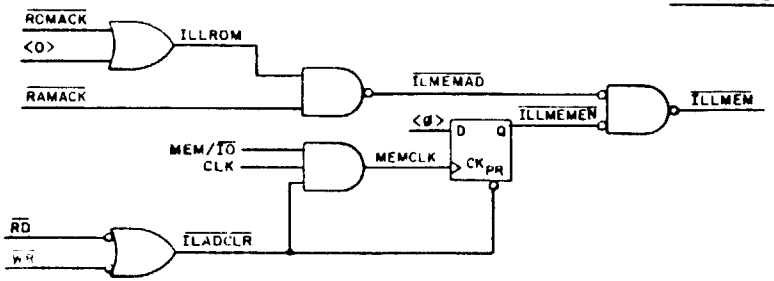
MEMORY CHIP SELECT DECODE







ILLEGAL MEMORY ADDRESS DETECTION



MEMORY DISABLE FOR END OF BUS CYCLE AND INTERRUPT CYCLE

ENGINEER J SELDIN / B BLOCK	DRAFTSMAN R K 9/25/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	MEMORY DECODING DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	6/4/87 DATE
	B-E7905	

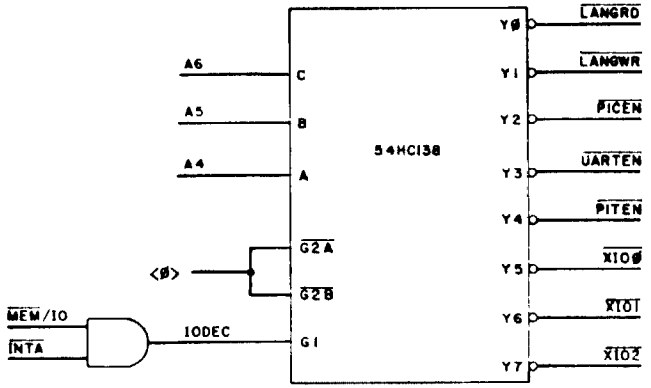
K&E 18 1153 2/86 11401

Last used R C D L

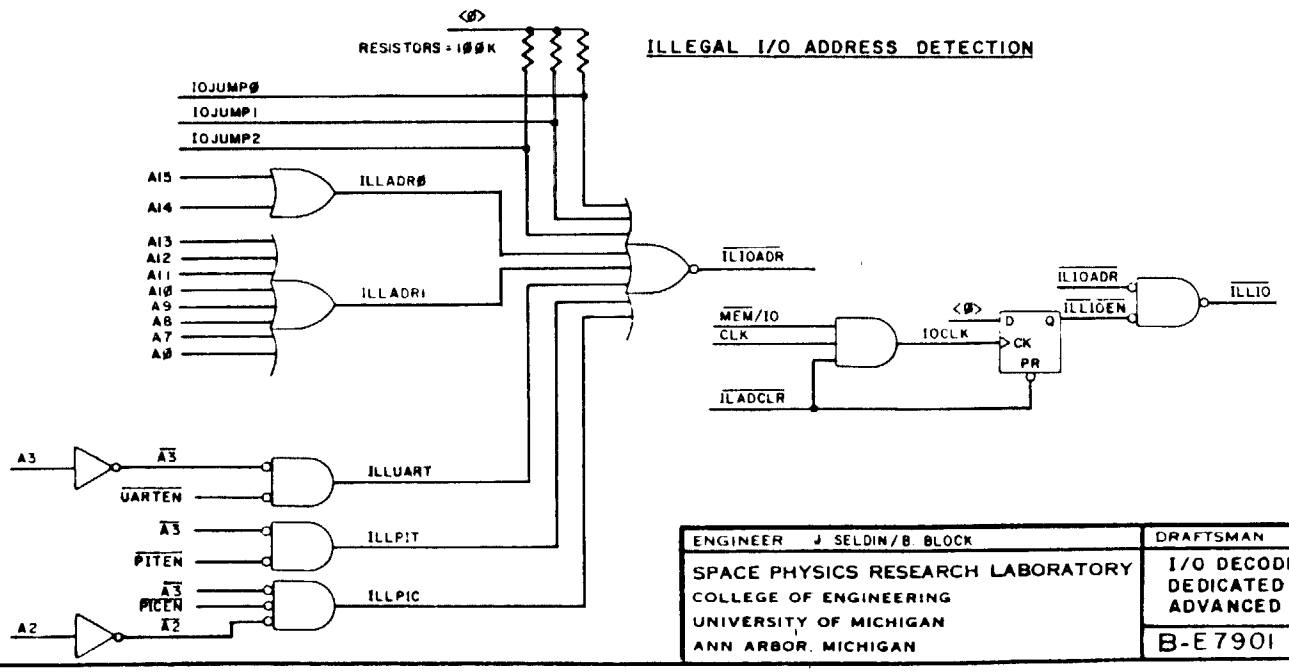
I/O ADDRESSES

LANGREAD:	ØØ – ØE
LANGWRITE:	1Ø – 1E
PIC:	2Ø, 22
UART:	3Ø, 32, 34, 36
PIT:	4Ø, 42, 44, 46
UNUSED:	5Ø – FFFF

I/O DECODE



ILLEGAL I/O ADDRESS DETECTION

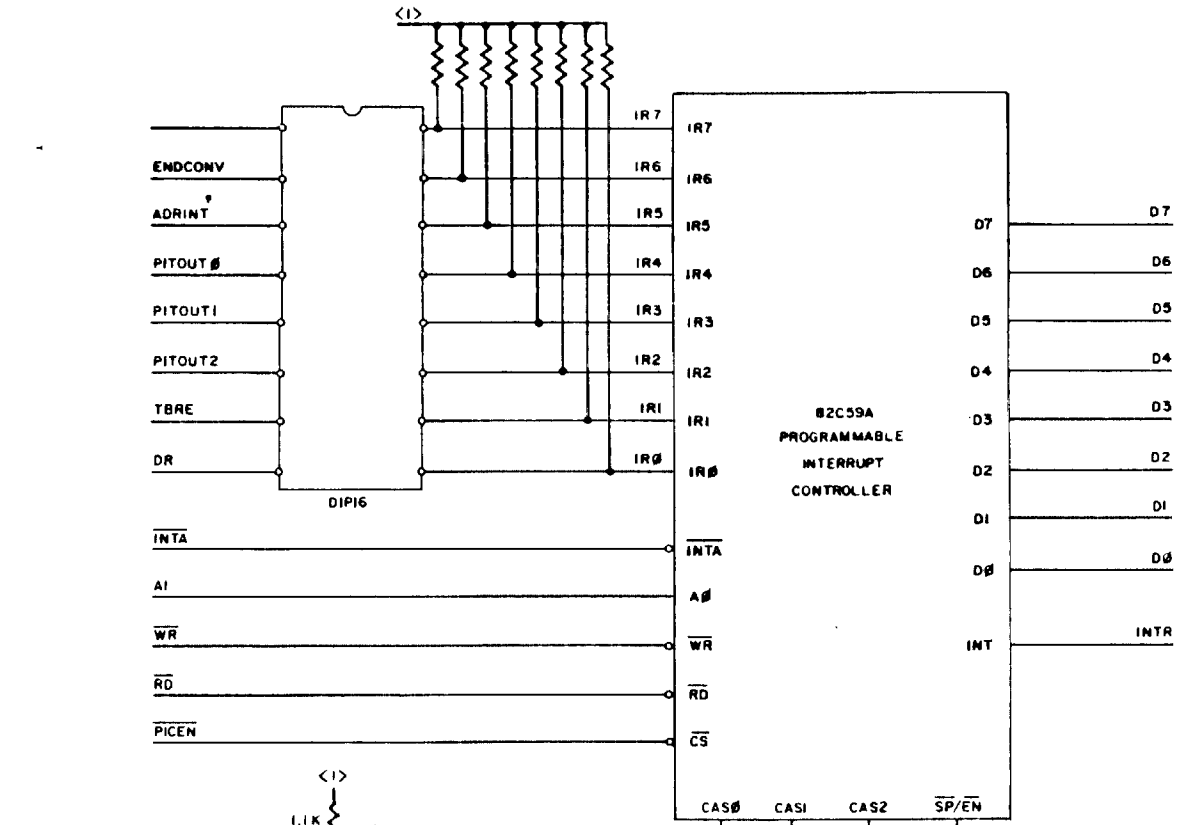


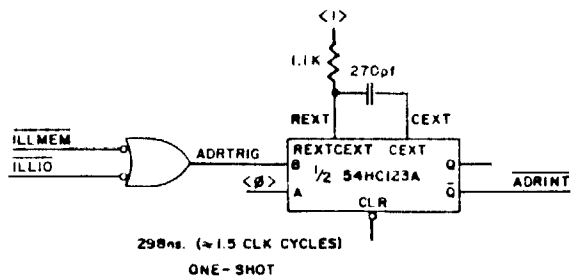
ENGINEER J. SELDIN/B. BLOCK	DRAFTSMAN R K 9/26/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	I/O DECODING DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	
	B-E7901	DATE

K&E 10 1153 2/86 1140F

LAST USED R C D L

PROGRAMMABLE INTERRUPT CONTROLLER (PIC)





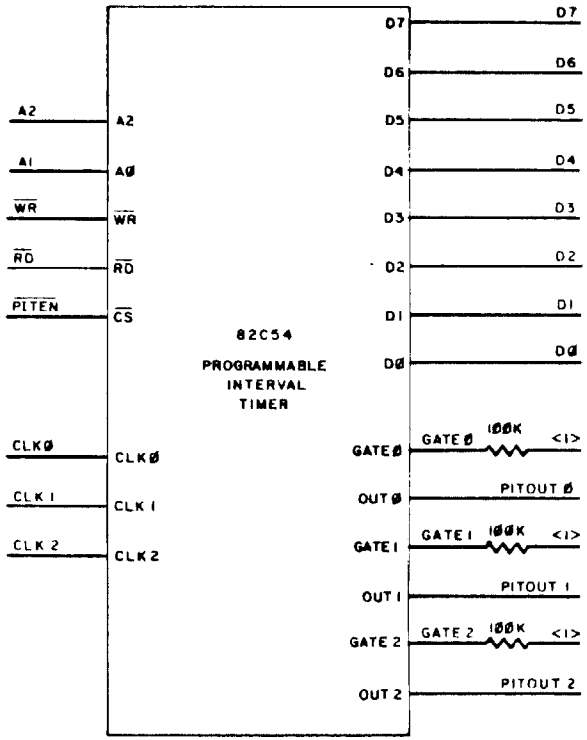
ILLEGAL ADDRESS INTERRUPT GENERATOR

ENGINEER J.SELDIN/B.BLOCK	DRAFTSMAN E.J./R.K. 10/7/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	PROG. INT. CONTROLLER DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	
	B-E7899	DATE

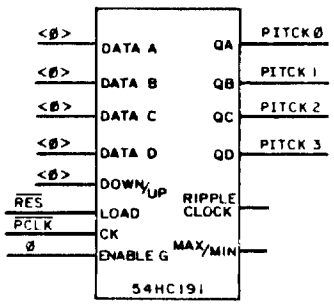
K&E 10 1153 2/86 1140

LAST USED R C D L

PROGRAMMABLE INTERVAL TIMER (PIT)

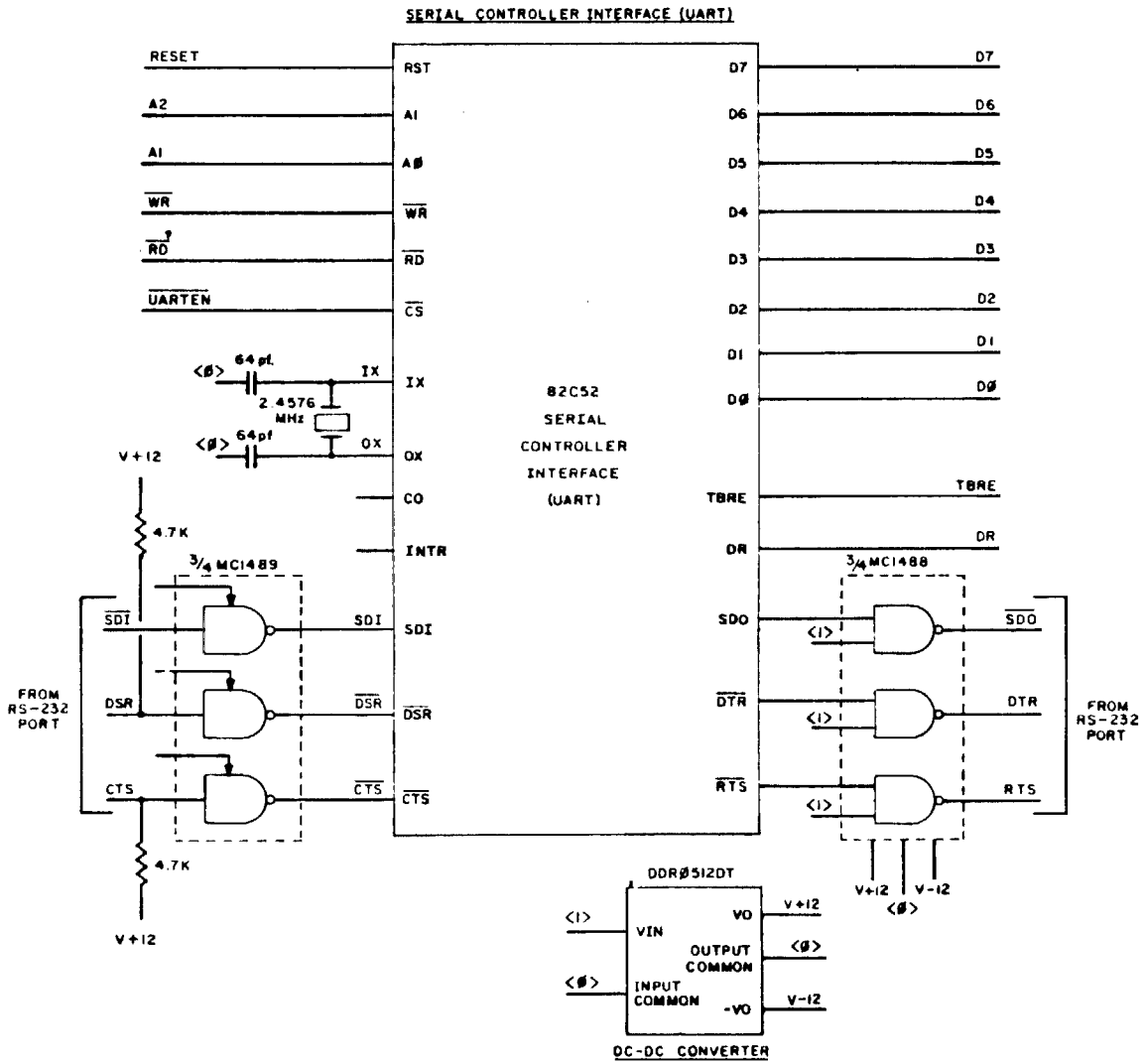


PIT CLOCK DIVIDER



PCLK	2.5 MHz
PITCKØ	1.25 MHz
PITCK1	625 KHz
PITCK2	312.5 KHz
PITCK3	156.25 KHz

ENGINEER B.P. BLOCK/J.H. SELDIN	DRAFTERS REK/EGJ 9/19/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	PROG. INTERVAL TIMER DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE B-E7904	DATE

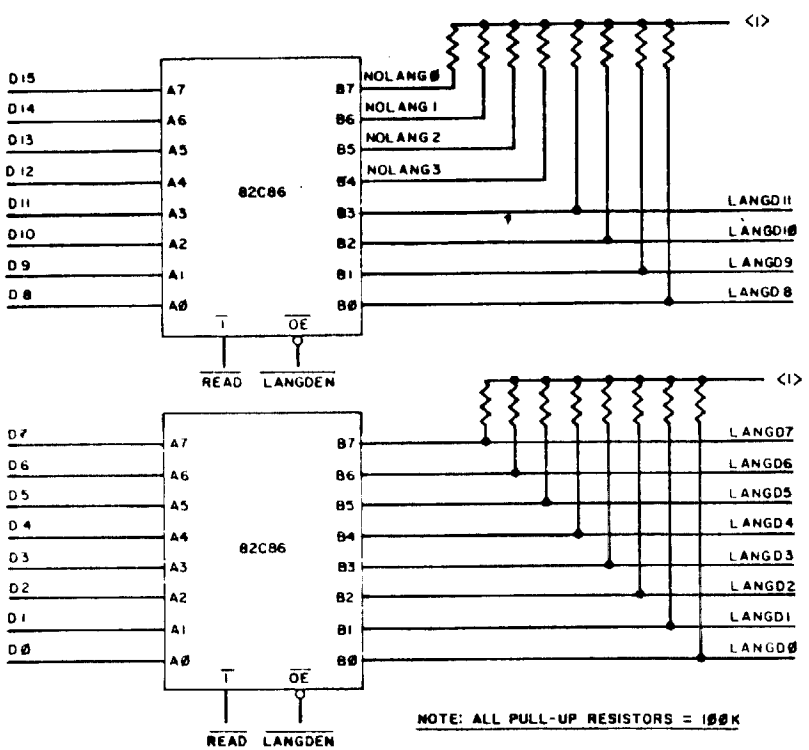


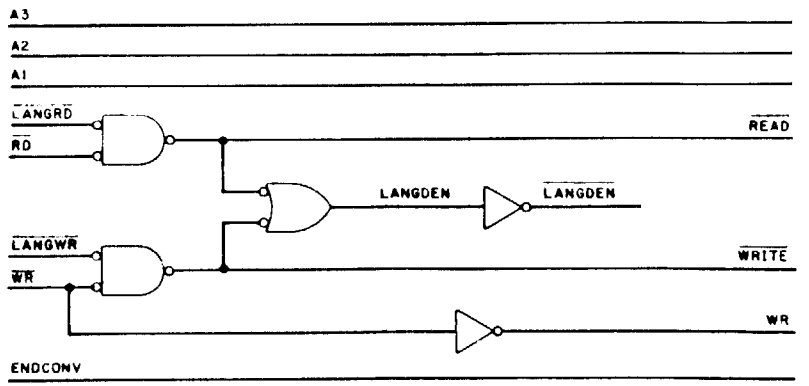
ENGINEER J.SELDIN/B.BLOCK	DRAFTSMAN R K 9/25/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	SERIAL CONTROLLE (UART) DEDICATED PROCESSOR ADVANCED LANGMUIR POBE	
	B-E7909	DATE

K&E 10 1153 2/86 1140*

LAST USED R C D L

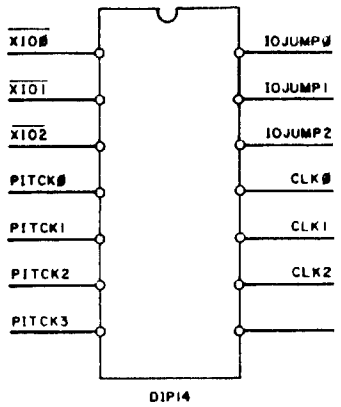
DE-LANG DATA/CONTROL TRANSCEIVERS

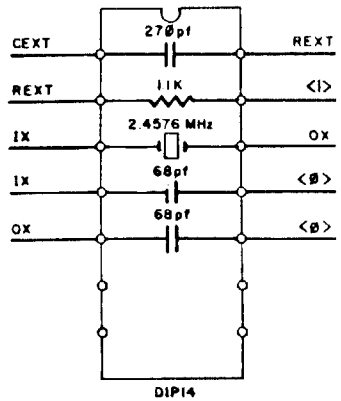




TO/FROM DE-LANG DIP CONNECTOR PORT

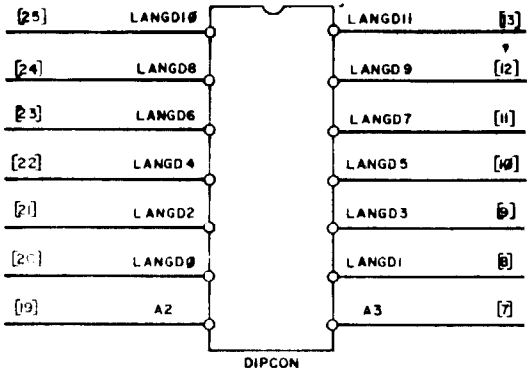
DIP HEADERS FOR DISCRETE DEVICES AND JUMPERS

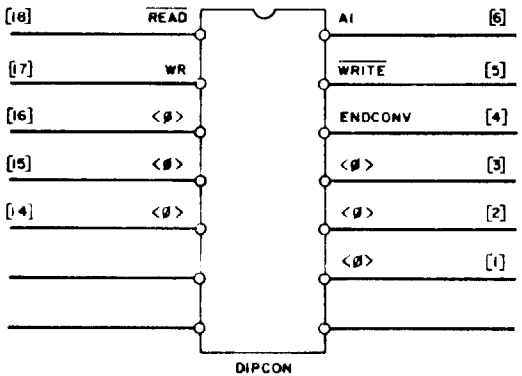




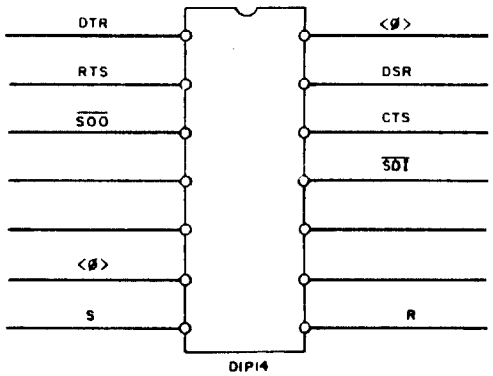
ENGINEER J SELDIN/B BLOCK	DRAFTSMAN R K 9/23/86
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR MICHIGAN	DE-LANG DATA/DIP HEADERS DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE B-E7907

DE-LANG DATA/CONTROL DIP CONNECTORS TO/FROM INTERFACE BOARD

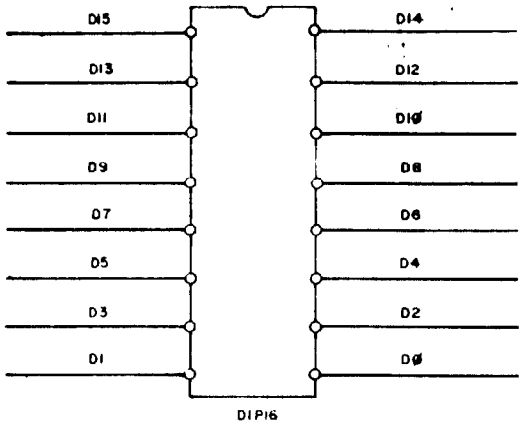


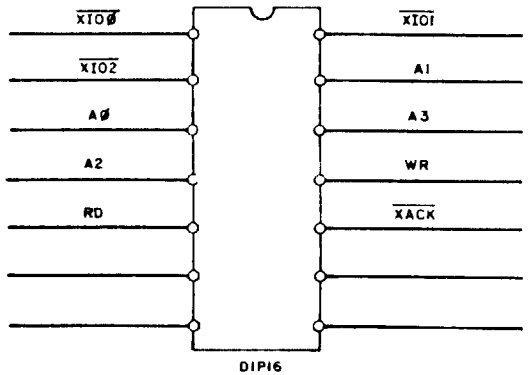


14 PIN DIP HEADER FOR RESET SWITCH TERMINALS AND RS232C SIGNALS



EXTERNAL I/O HEADERS: 16-BIT DATA BUS, EXTERNAL I/O SELECT, ADDRESS LINES AND READ/WRITE CONTROL

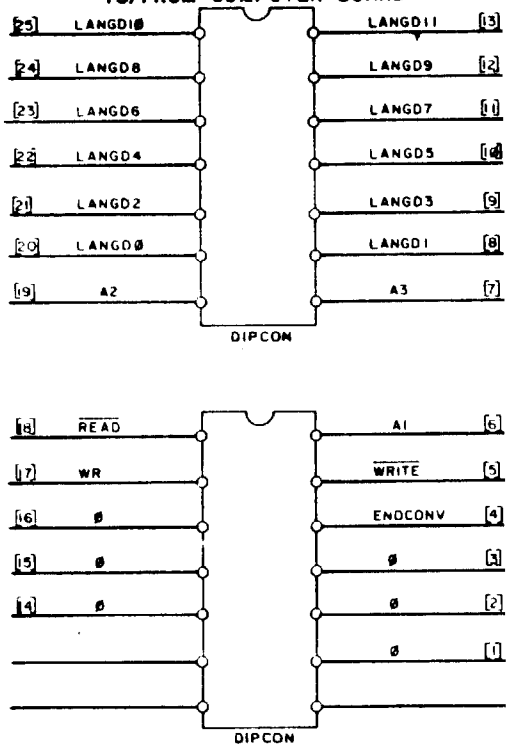




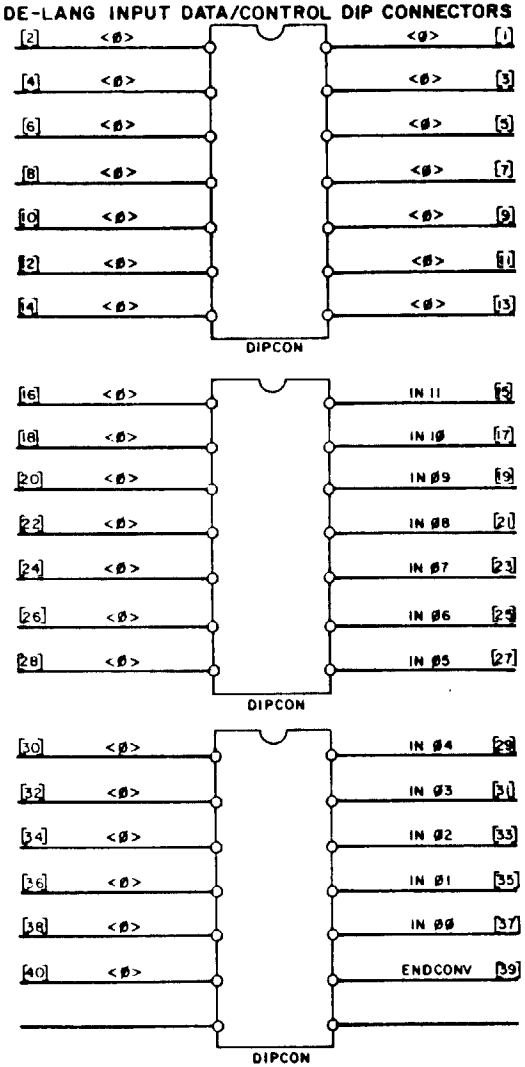
ENGINEER J.H SELDIN/ B.P.BLOCK	DRAFTSMAN R K 9/19/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	I/O CONNECTORS/DIPS DEDICATED PROCESSOR ADVANCED LANGMUIR PROBE	
	B-E7906	DATE

DE-LANG I/O CONNECTOR PORTS

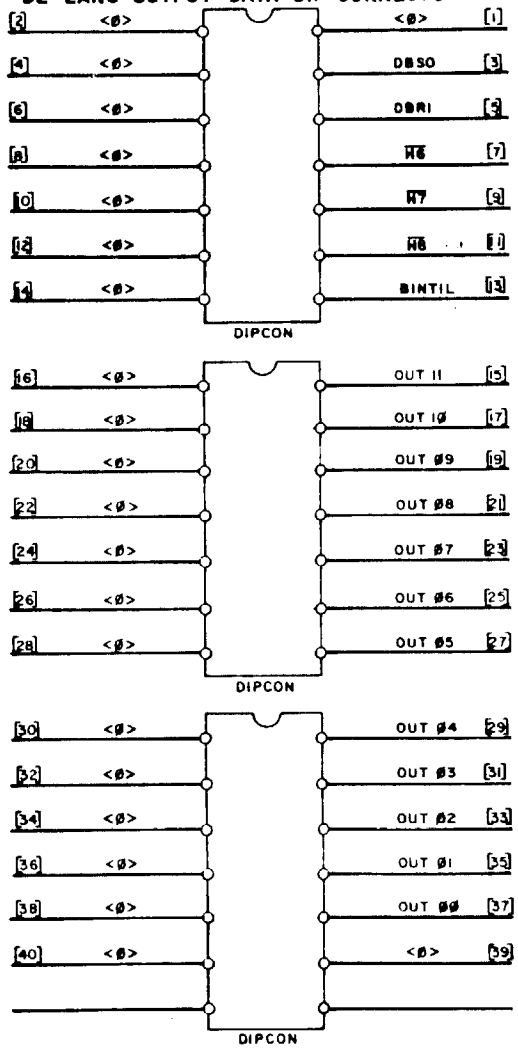
DE-LANG DATA/CONTROL DIP CONNECTORS TO/FROM COMPUTER BOARD



DE-LANG INPUT DATA/CONTROL DIP CONNECTORS



DE-LANG OUTPUT DATA DIP CONNECTORS



NOTE:
CONNECTOR PIN NUMBERS ARE INDICATED IN BRACKETS NEXT TO SIGNAL NAMES.

ENGINEER B.P. BLOCK/J.H. SELDIN	DRAFTERS REK/EGJ 9/18/86
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	I/O CONNECTOR PORTS PROCESSOR INTERFACE ADVANCED LANGMUIR PROBE
	B-E7898

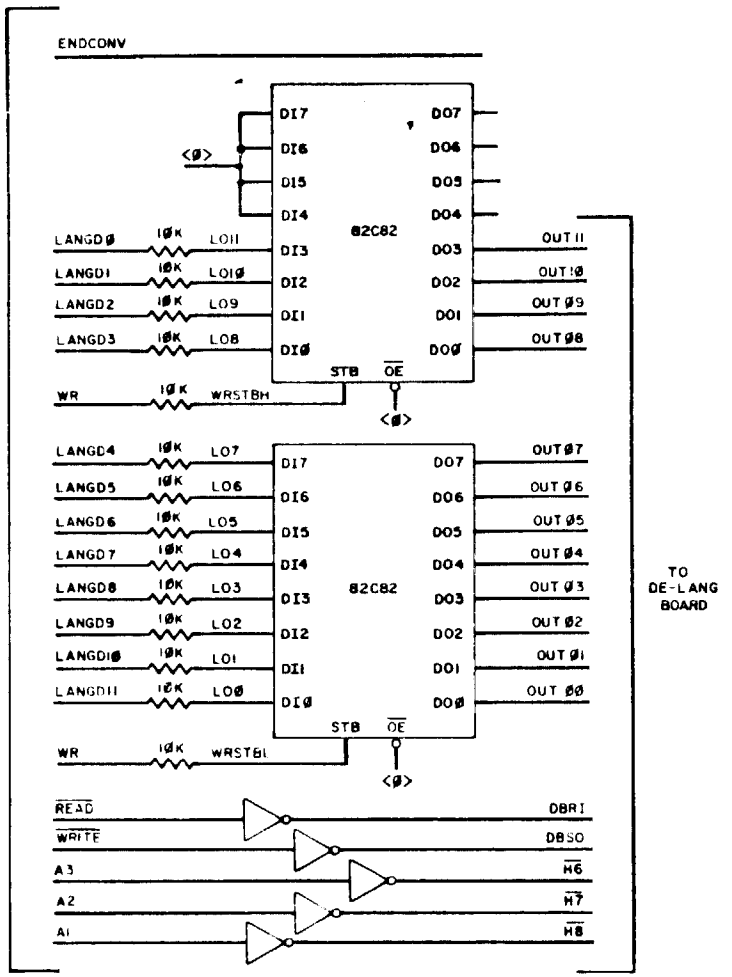
DATE

K&E 10 1153 2/86 1140P

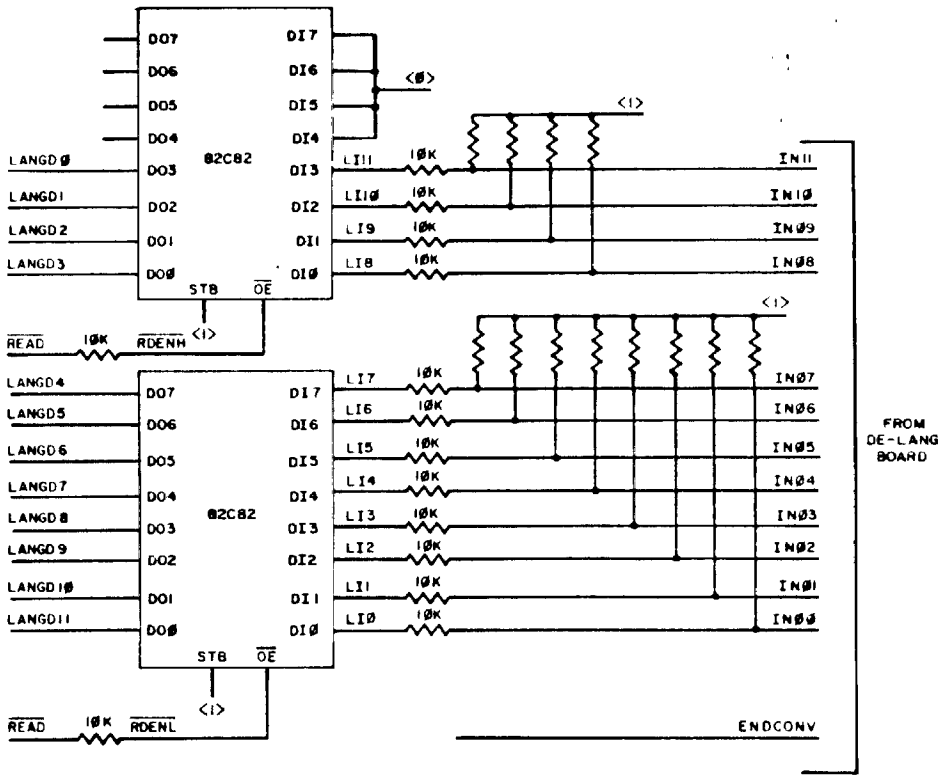
LAST USED R C D L

DE-LANG DATA/CONTROL DRIVERS

DATA TO DE-LANG



DATA FROM DE-LANG



NOTE: ALL PULL-UP RESISTORS=IØØK

ENGINEER J SELDIN B BLOCK	DRAFTSMAN R K 9/23/86	
SPACE PHYSICS RESEARCH LABORATORY COLLEGE OF ENGINEERING UNIVERSITY OF MICHIGAN ANN ARBOR, MICHIGAN	DE-LANG DATA DRIVERS PROCESSOR INTERFACE ADVANCED LANGMUIR PROBE B-E 7900	DATE

```
                    $TITLE
                                    *** DE-LANG CONTROLLER COMPUTER BOARD ***
                                            *** VERSION 1 ***
                    *
                    *
                    *           WIREWRAP DEVICE DESCRIPTION:  JULY 22 1986, JOHN SELDIN
                    *
                    *
                    *
                    $SET CONTCHAR=% BAYS=4 ELEMAX=500 NETMAX=1000 ECOMAX=500 WIRES=OFF PUNCH=OFF ECOS=ON
                    $SET HI=PWR LO=GND
                    $DATA
                    *
                    *
                    $SET ECHO=OFF
                    *
                    *
                    $EJECT
```

```
 80                          *
 81                          *           DATA/ADDRESS BUS DIVISION:  ADDRESS LATCHES AND DATA TRANSCEIVERS
 82                          *           DWG B-E???? (1.2)
 83                          *
 84                          *
 85                          *
 86                          *     ADDRESS LATCHES:  82C82
 87                          *
 88       E16  A-A-1         82C82 DI=<1>,<1>,<1>,<1>,ADR19,ADR18,ADR17,ADR16 DO=,,,,A19,A18,A17,A16 STB=ALE OE-=<0>
 89       E17  A-A-1         82C82 DI=AD15,AD14,AD13,AD12,AD11,AD10,AD9,AD8 DO=A15,A14,A13,A12,A11,A10,A9,A8 STB=ALE OE-=<0>
 90       E18  A-A-1         82C82 DI=AD7,AD6,AD5,AD4,AD3,AD2,AD1,AD0 DO=A7,A6,A5,A4,A3,A2,A1,A0 STB=ALE OE-=<0>
 91                          *
 92                          *
 93                          *     DATA TRANSCEIVERS:  82C86
 94                          *
 95       E19  A-A-1         82C86 A=AD15,AD14,AD13,AD12,AD11,AD10,AD9,AD8 B=D15,D14,D13,D12,D11,D10,D9,D8 T=DT/R- OE-=DEN-
 96       E20  A-A-1         82C86 A=AD7,AD6,AD5,AD4,AD3,AD2,AD1,AD0 B=D7,D6,D5,D4,D3,D2,D1,D0 T=DT/R- OE-=DEN-
 97                          *
 98                          *
 99                          *     DATA/ADDRESS BUS PULL-UP RESISTORS (100K)
100                          *
101       E21  A-A-0         PULLUP R=ADR19
102       E22  A-A-0         PULLUP R=ADR18
103       E23  A-A-0         PULLUP R=ADR17
104       E24  A-A-0         PULLUP R=ADR16
105       E25  A-A-0         PULLUP R=AD15
106       E26  A-A-0         PULLUP R=AD14
107       E27  A-A-0         PULLUP R=AD13
108       E28  A-A-0         PULLUP R=AD12
109       E29  A-A-0         PULLUP R=AD11
110       E30  A-A-0         PULLUP R=AD10
111       E31  A-A-0         PULLUP R=AD9
112       E32  A-A-0         PULLUP R=AD8
113       E33  A-A-1         PULLUP R=AD7
114       E34  A-A-1         PULLUP R=AD6
115       E35  A-A-1         PULLUP R=AD5
116       E36  A-A-1         PULLUP R=AD4
117       E37  A-A-1         PULLUP R=AD3
118       E38  A-A-1         PULLUP R=AD2
119       E39  A-A-1         PULLUP R=AD1
120       E40  A-A-1         PULLUP R=AD0
121                          *
122                          *
123                          *     DATA BUS PULL-UP RESISTORS (100K)
124                          *
125       E41  A-A-1         PULLUP R=D15
126       E42  A-A-1         PULLUP R=D14
127       E43  A-A-1         PULLUP R=D13
128       E44  A-A-1         PULLUP R=D12
129       E45  A-A-1         PULLUP R=D11
130       E46  A-A-1         PULLUP R=D10
131       E47  A-A-1         PULLUP R=D9
132       E48  A-A-1         PULLUP R=D8
133       E49  A-A-1         PULLUP R=D7
134       E50  A-A-1         PULLUP R=D6
```

```
144                     *
145                     *           MEMORY ADDRESS/CHIP-SELECT DECODING AND ILLEGAL ADDRESS DETECTION
146                     *           DWG B-E???? (1.3)
147                     *
148                     *
149                     *
150                     *     MEMORY ADDRESS DECODE
151                     *
152       E57  A-A-1    HC4078 I=A19,A18,A17,A16,A15,A14,A13,A12 OR=RAMADR-
153       E58  A-A-1    HC30 I=<1>,<1>,<1>,A19,A18,A17,A16,A15 O=ROMADR-
154       E59  A-A-0    HC04 I=MEM/IO- O=MEM-/IO
155       E60  A-A-0    HC04 I=A14 O=A14-
156       E61  A-A-1    HC32 I=RAMADR-,MEM-/IO O=RAMACK-
157       E62  A-A-1    HC32 I=ROMADR-,MEM-/IO O=ROMACK-
158       E63  A-A-1    HC32 I=A14,ROMACK- O=ROMLOW-
159       E64  A-A-1    HC32 I=A14-,ROMACK- O=ROMHIGH-
160                     *
161                     *
162                     *     MEMORY DISABLE FOR END OF BUS CYCLE AND INTERRUPT CYCLE
163                     *
164       E65  A-A-2    HC11 I=RD-,WR-,INTA- O=CYCSET-
165       E66  A-A-2    HC74 D=<1> CK=ALE CLR=CYCSET- PR=<1> Q=EOCEN-
166       E67  A-A-2    HC32 I=EOCEN-,ILADCLR- O=EOCSET-
167       E68  A-A-2    HC74 D=<1> CK=ALE- CLR=EOCSET- PR=<1> Q=MEMEN-
168       E69  A-A-2    HC02 I=MEMEN-,ILADCLR- O=MEMDIS
169                     *
170                     *
171                     *     MEMORY CHIP SELECT DECODE
172                     *
173       E70  A-A-2    HC4075 I=ROMHIGH-,A0,MEMDIS O=ROMHEVN-
174       E71  A-A-2    HC4075 I=ROMHIGH-,BHE-,MEMDIS O=ROMHODD-
175       E72  A-A-2    HC4075 I=ROMLOW-,A0,MEMDIS O=ROMLEVN-
176       E73  A-A-2    HC4075 I=ROMLOW-,BHE-,MEMDIS O=ROMLODD-
177       E74  A-A-2    HC4075 I=RAMACK-,A0,MEMDIS O=RAMEVEN-
178       E75  A-A-2    HC4075 I=RAMACK-,BHE-,MEMDIS O=RAMODD-
179                     *
180                     *
181                     *     ILLEGAL MEMORY ADDRESS DETECTION
182                     *
183       E76  A-A-2    HC32 I=ROMACK-,RD- O=ILLROM
184       E77  A-A-0    HC00 I=RD-,WR- O=ILADCLR-
185       E78  A-A-2    HC00 I=ILLROM,RAMACK- O=ILMEMAD-
186       E79  A-A-2    HC11 I=MEM/IO-,CLK,ILADCLR- O=MEMCLK
187       E80  A-A-2    HC74 D=<0> CK=MEMCLK CLR=<1> PR=ILADCLR- Q=ILMEMEN-
188       E81  A-A-2    HC32 I=ILMEMEN-,ILMEMAD- O=ILLMEM-
189                     *
190                     *
191                     $EJECT
```

```
                    *
                    *                    I/O DECODING AND ILLEGAL ADDRESS DETECTION
                    *                    DWG B-E???? (1.5)
                    *
                    *
                    *
                    *     I/O DECODE
                    *
      E88  A-A-3    HC138 SEL=A4,A5,A6 EN=<1>,<1>,IODEC O=XIO2-,XIO1-,XIO0-,PITEN-,UARTEN-,PICEN-,LANGWR-,LANGRD-
      E89  A-A-0    HC08 I=MEM-/IO,INTA- O=IODEC
                    *
                    *
                    *     ILLEGAL I/O ADDRESS DETECTION
                    *
      E90  A-A-2    HC32 I=A15,A14 O=ILLADR0
      E91  A-A-3    HC4078 I=A13,A12,A11,A10,A9,A8,A7,A0 OR=ILLADR1
      E92  A-A-0    HC04 I=A3 O=A3-
      E93  A-A-3    HC04 I=A2 O=A2-
      E94  A-A-2    HC02 I=A3-,UARTEN- O=ILLUART
      E95  A-A-2    HC02 I=A3-,PITEN- O=ILLPIT
      E96  A-A-3    HC27 I=A3-,A2-,PICEN- O=ILLPIC
      E97  A-A-3    HC4078 I=IOJUMP0,IOJUMP1,IOJUMP2,ILLADR0,ILLADR1,ILLUART,ILLPIT,ILLPIC NOR=ILIOADR-
      E98  A-A-3    PULLDOWN R=IOJUMP0
      E99  A-A-3    PULLDOWN R=IOJUMP1
     E100  A-A-3    PULLDOWN R=IOJUMP2
     E101  A-A-2    HC11 I=MEM-/IO,CLK,ILADCLR- O=IOCLK
     E102  A-A-2    HC74 D=<0> CK=IOCLK CLR=<1> PR=ILADCLR- Q=ILLIOEN-
     E103  A-A-4    HC32 I=ILLIOEN-,ILIOADR- O=ILLIO-
                    *
                    *
                    $EJECT
```

```
305                     *
306                     *               PROGRAMMABLE INTERVAL TIMER AND PERIPHERAL CLOCK DIVIDER
307                     *               DWG B-E???? (1.7)
308                     *
309                     *
310                     *
311                     *     PROGRAMMABLE INTERVAL TIMER:  82C54
312                     *
314     E116  A-A-4     82C54 D=D7,D6,D5,D4,D3,D2,D1,D0 CK=CLK0,CLK1,CLK2 OUT=PITOUT0,PITOUT1,PITOUT2 GATE=GATE0,GATE1,GATE2
                              ADR=A2,A1 CS-=PITEN- RD-=RD- WR-=WR-
315                     *
316                     *     *
317                     *     GATE INPUT PULL-UP RESISTORS (100K)
318                     *
319     E117  A-A-4     PULLUP R=GATE0
320     E118  A-A-4     PULLUP R=GATE1
321     E119  A-A-4     PULLUP R=GATE2
322                     *
323                     *
324                     *      PERIPHERAL CLOCK DIVIDER
325                     *
326     E120  A-A-4     HC191 CK=PCLK U/D=<0> LD=RES- EN=<0> DATA=<0>,<0>,<0>,<0> Q=PITCK0,PITCK1,PITCK2,PITCK3
327                     *
328                     *
329                     $EJECT
```

```
362                     *
363                     *     DE-LANG DATA/CONTROL TRANSCEIVER INTERFACE AND DIP HEADERS FOR DISCRETE DEVICES AND JUMPERS
364                     *     DWG B-E???? (1.9)
365                     *
366                     *
367                     *
368                     *      DE-LANG DATA/CONTROL TRANSCEIVER
369                     *
370                     *
371                     *        TRANSCEIVERS
372                     *
374       E129  A-B-0   82C86 A=D15,D14,D13,D12,D11,D10,D9,D8 B=NOLANGO,NOLANG1,NOLANG2,NOLANG3,LANGD11,LANGD10,LANGD9,LANGD8
                              T=READ- OE-=LANGDEN-
376       E130  A-B-0   82C86 A=D7,D6,D5,D4,D3,D2,D1,DO B=LANGD7,LANGD6,LANGD5,LANGD4,LANGD3,LANGD2,LANGD1,LANGDO T=READ-
                              OE-=LANGDEN-
377                     *
378                     *
379                     *        DE-LANG DATA PULL-UP RESISTORS (100K)
380                     *
381       E131  A-A-4   PULLUP R=NOLANGO
382       E132  A-A-4   PULLUP R=NOLANG1
383       E133  A-A-4   PULLUP R=NOLANG2
384       E134  A-A-4   PULLUP R=NOLANG3
385       E135  A-A-4   PULLUP R=LANGD11
386       E136  A-A-4   PULLUP R=LANGD10
387       E137  A-A-4   PULLUP R=LANGD9
388       E138  A-A-4   PULLUP R=LANGD8
389       E139  A-A-4   PULLUP R=LANGD7
390       E140  A-A-4   PULLUP R=LANGD6
391       E141  A-B-0   PULLUP R=LANGD5
392       E142  A-B-0   PULLUP R=LANGD4
393       E143  A-B-0   PULLUP R=LANGD3
394       E144  A-B-0   PULLUP R=LANGD2
395       E145  A-B-0   PULLUP R=LANGD1
396       E146  A-B-0   PULLUP R=LANGDO
397                     *
398                     *
399                     *        DE-LANG CONTROL SIGNAL GENERATION
400                     *
401       E147  A-A-4   HC32 I=LANGRD-,RD- O=READ-
402       E148  A-A-4   HC32 I=LANGWR-,WR- O=WRITE-
403       E149  A-A-2   HC00 I=READ-,WRITE- O=LANGDEN
404       E150  A-A-3   HC04 I=LANGDEN O=LANGDEN-
405       E151  A-A-3   HC04 I=WR- O=WR
406                     *
407                     *
408                     *
409                     *      2 14-PIN DIP HEADERS FOR DISCRETE DEVICES AND JUMPERS
410                     *
411       E152  A-B-0   DIP14 I17=XIOO-,XIO1-,XIO2-,PITCKO,PITCK1,PITCK2,PITCK3 I148=IOJUMPO,IOJUMP1,IOJUMP2,CLKO,CLK1,CLK2,
412       E153  A-B-0   DIP14 I17=CEXT,REXT,IX,IX,OX,, I148=REXT,<1>,OX,<O>,<O>,,
413                     *
414                     *
415                     $EJECT
```

```
458                     *                                       ECO LISTING
459                     *
460                     *
461                     $ECO 9-10-86
462                     *
463                     *     MODIFICATION OF I/O DECODING ENABLES ON HC138
464                     *
465                     $MODIFY E88 SEL=A4,A5,A6 EN=<0>,<0>,IODEC O=XIO2-,XIO1-,XIO0-,PITEN-,UARTEN-,PICEN-,LANGWR-,LANGRD-
466                     *
467                     *
468                     *      *
469                     $ECO 9-16-86
470                     *
471                     *     BUFFERING OF NOISY CLOCK SIGNAL OUT OF 82C84
472                     *
473     E159  A-A-3     HC04 I=CLK- O=CK
474                     *
477                     $MODIFY E1 A=ADR19,ADR18,ADR17,ADR16 AD=AD15,AD14,AD13,AD12,AD11,AD10,AD9,AD8,AD7,AD6,AD5,AD4,AD3,AD2,AD
                            1,ADO NMI=NMI INT=INTR CK=CK BHE-=BHE- MN/MX-=<1> R-=RD- HLD=<0> W-=WR- M/I-=MEM/IO- DATCTL=DT/R-,DE
                            N- ALE=ALE INTA-=INTA- TST-=<0> RDY=READY RST=RESET
478                     *
479                     *
479.1                   *
479.2                   $ECO 10-10-86
479.3                   *
479.4                   *     PULL-UPS FOR RS-232 RECEIVER BUFFERS (FOR MINIMUM HAND-SHAKING)
479.5                   *
479.6   E160  A-B-0     DIP14 I17=V+12,V+12,,,,,, I148=DSR,CTS,,,,,,
479.7                   *
479.8                   *
480                     $END
```

0 ERROR(S) HAVE BEEN DETECTED

CTS	358,439,479.6
CTS-	339,358
CYCSET-	164,165
DEN-	37,95,96,477
DR	283,339
DSR	357,439,479.6
DSR-	339,357
DT/R-	37,95,96,477
DTR	350,439
DTR-	339,350
D0	96,140,209,221,234,278,314,339,376,446
D1	96,139,209,221,234,278,314,339,376,446
D10	95,130,203,215,229,374,446
D11	95,129,203,215,229,374,446
D12	95,128,203,215,229,374,446
D13	95,127,203,215,229,374,446
D14	95,126,203,215,229,374,446
D15	95,125,203,215,229,374,446
D2	96,138,209,221,234,278,314,339,376,446
D3	96,137,209,221,234,278,314,339,376,446
D4	96,136,209,221,234,278,314,339,376,446
D5	96,135,209,221,234,278,314,339,376,446
D6	96,134,209,221,234,278,314,339,376,446
D7	96,133,209,221,234,278,314,339,376,446
D8	95,132,203,215,229,374,446
D9	95,131,203,215,229,374,446
ENDCONV	283,432
EOCEN-	165,166
EOCSET-	166,167
FEEDBACK	73,74
GATE0	314,319
GATE1	314,320
GATE2	314,321
GND12	344,349,439
H001#	37,37,37,477,477
H002#	42,42
H003#	48,48
H004#	58,58,58
H005#	59,59
H006#	60,60
H007#	61,61
H008#	66,66,66
H009#	66,66
H010#	88,88,89
H011#	88,88,88,88,88
H012#	90,90
H013#	95,95,153,153,153,153
H014#	96,96
H015#	113,113
H016#	132,132,165,165
H017#	152,152
H018#	156,156
H019#	164,164
H020#	165,165,167,167,187
H021#	166,166,203

NET NAME DICTIONARY

NET		LD	DR	PINS
CTS		1	0	D-B-10 ,C-G-45 ,A-U-42
CTS-		1	10	C-G-12 ,C-G-47
CYCSET-		1	10	A-J-25 ,B-A-19
DEN-		2	0	A-C-15 ,A-F-09 ,A-G-20
DR		0	10	C-C-08 ,C-G-03
DSR		1	0	D-B-09 ,C-F-44 ,A-U-41
DSR-		1	10	C-F-46 ,C-G-11
DT/R-		2	0	A-C-14 ,A-G-10 ,A-H-21
DTR		0	10	C-E-21 ,A-U-17
DTR-		1	10	C-E-19 ,C-G-10
DO	0	1	10	B-G-22 ,C-D-35 ,C-A-44 ,B-E-41 ,A-H-39 ,A-H-13 ,B-D-11 ,B-U-31 ,C-E-03 ,C-H-01
D1	0	1	10	B-U-08 ,B-G-23 ,B-D-12 ,A-H-14 ,A-G-39 ,B-E-42 ,C-A-43 ,C-D-34 ,C-E-04 ,C-H-02
D10	0	1	10	C-G-31 ,B-B-45 ,A-G-04 ,A-H-25 ,B-D-28 ,B-G-11 ,B-U-26
D11	0	1	10	C-G-32 ,B-D-46 ,B-F-29 ,A-H-26 ,A-G-05 ,B-J-12 ,B-U-03
D12	0	1	10	C-G-33 ,B-D-45 ,A-G-06 ,A-H-27 ,B-F-28 ,B-J-11 ,B-U-25
D13	0	1	10	C-G-34 ,B-D-44 ,A-G-07 ,A-H-28 ,B-F-27 ,B-J-10 ,B-U-02
D14	0	1	10	A-G-08 ,A-H-29 ,B-D-43 ,B-F-26 ,B-J-09 ,B-U-24 ,C-G-35
D15	0	1	10	A-G-09 ,A-H-30 ,B-D-42 ,B-F-25 ,B-J-08 ,B-U-01 ,C-G-36
D2	0	1	10	A-G-38 ,A-H-15 ,B-D-13 ,B-G-24 ,B-E-43 ,C-A-42 ,C-D-33 ,C-E-05 ,C-H-03 ,B-U-30
D3	0	1	10	A-G-37 ,A-H-16 ,B-U-07 ,B-F-14 ,B-J-25 ,B-G-44 ,C-A-41 ,C-D-32 ,C-E-06 ,C-H-04
D4	0	1	10	A-G-36 ,A-H-17 ,B-U-29 ,B-F-13 ,B-J-24 ,B-G-43 ,C-A-40 ,C-D-31 ,C-E-07 ,C-H-05
D5	0	1	10	A-G-35 ,A-H-18 ,B-U-06 ,B-F-12 ,B-J-23 ,B-G-42 ,C-A-39 ,C-D-30 ,C-E-08 ,C-H-06
D6	0	1	10	A-G-34 ,A-H-19 ,B-U-28 ,B-F-11 ,B-J-22 ,B-G-41 ,C-A-38 ,C-D-29 ,C-E-09 ,C-H-07
D7	0	1	10	A-G-33 ,A-H-20 ,B-U-05 ,B-F-10 ,B-J-21 ,B-G-40 ,C-A-37 ,C-D-28 ,C-E-10 ,C-H-08
D8	0	1	10	A-G-02 ,A-G-32 ,B-B-43 ,B-D-26 ,B-G-09 ,B-U-27 ,C-G-29
D9	0	1	10	B-U-04 ,B-G-10 ,A-G-03 ,A-H-24 ,B-D-27 ,B-B-44 ,C-G-30
ENDCONV		0	0	C-C-02 ,A-U-34
EOCEN-		1	10	A-J-29 ,A-J-33
EOCSET-		1	10	A-J-35 ,B-A-26
FEEDBACK		1	10	A-C-43 ,A-D-45
GATE0		2	0	C-C-12 ,C-D-38
GATE1		2	0	C-C-13 ,C-F-38
GATE2		2	0	C-C-14 ,C-F-36
GND12		1	0	A-U-40 ,C-E-22 ,C-G-24
HOO1#		1	0	A-C-01 ,A-C-08 ,A-D-V08
HOO2#		0	0	A-B-22 ,A-B-V24
HOO3#		0	0	A-B-30 ,A-B-V32
HOO4#		1	0	A-C-22 ,A-D-22 ,A-D-V24
HOO5#		0	0	A-D-30 ,A-D-V32
HOO6#		0	0	A-D-39 ,A-D-V40
HOO7#		0	0	A-E-01 ,A-F-V08
HOO8#		1	0	A-E-09 ,A-E-16 ,A-F-V16
HOO9#		1	0	A-D-10 ,A-D-V16
HO10#		1	0	A-E-25 ,A-F-29 ,A-F-V32
HO11#		3	0	A-F-18 ,A-E-22 ,A-E-23 ,A-E-24 ,A-F-V24
HO12#		0	0	A-F-40 ,A-F-V40
HO13#		3	0	A-G-01 ,A-H-01 ,A-H-02 ,A-H-03 ,A-J-01 ,A-H-V08
HO14#		0	0	A-H-12 ,A-H-V16
HO15#		0	0	A-H-23 ,A-H-V24
HO16#		2	0	A-H-32 ,A-H-V32 ,A-J-28 ,A-J-26
HO17#		0	0	A-H-41 ,A-H-V48
HO18#		0	0	B-A-09 ,B-B-V16
HO19#		0	0	B-A-17 ,B-B-V24
HO20#		3	0	B-B-25 ,B-A-25 ,B-A-27 ,B-A-29 ,B-B-V32
HO21#		0	0	B-A-33 ,B-B-33 ,B-B-V40

NET	LD	DR	PINS
IR4	2	O	A-H-35 ,C-C-40 ,C-D-04
IR5	2	O	A-H-36 ,C-C-39 ,C-D-03
IR6	2	O	A-H-37 ,C-C-38 ,C-D-02
IR7	2	O	A-H-38 ,C-C-37 ,C-D-01
IX	1	O	D-A-03 ,D-A-04 ,C-E-13
LANGDEN	1	10	B-C-23 ,B-H-46
LANGDEN-	2	10	B-H-47 ,C-G-37 ,C-H-09
LANGD0	1	O	C-H-45 ,C-J-02 ,A-U-06
LANGD1	1	O	C-H-44 ,C-J-03 ,A-U-29
LANGD10	1	O	A-U-01 ,C-D-15 ,C-H-32
LANGD11	1	O	A-U-24 ,C-D-16 ,C-H-33
LANGD2	1	O	C-H-43 ,C-J-04 ,A-U-05
LANGD3	1	O	C-H-42 ,C-J-05 ,A-U-28
LANGD4	1	O	C-H-41 ,C-J-06 ,A-U-04
LANGD5	1	O	C-H-40 ,C-J-07 ,A-U-27
LANGD6	1	O	A-U-03 ,C-D-11 ,C-J-08
LANGD7	1	O	A-U-26 ,C-D-12 ,C-J-09
LANGD8	1	O	A-U-02 ,C-D-13 ,C-H-30
LANGD9	1	O	A-U-25 ,C-D-14 ,C-H-31
LANGRD-	2	10	A-C-39 ,B-J-28 ,C-A-29
LANGRDY	1	10	A-C-41 ,A-D-01
LANGWR-	2	10	A-C-40 ,B-J-29 ,C-B-31
LO01#	1	O	A-D-16 ,A-D-G12,A-C-10
LO02#	1	O	A-C-18 ,A-D-G20
LO03#	0	O	A-A-01 ,A-B-G04
LO04#	0	O	A-A-20 ,A-B-G20
LO05#	1	O	A-B-G28,A-A-28 ,A-A-30
LO06#	2	O	A-B-33 ,A-B-35 ,A-B-G36,A-A-38
LO07#	0	O	A-B-42 ,A-B-G44,A-B-45 ,A-B-46
LO08#	0	O	A-C-28 ,A-D-G28
LO09#	0	O	A-C-45 ,A-D-G44
LO10#	0	O	A-U-11 ,A-U-12 ,A-U-35 ,A-D-G04,A-D-07
LO11#	1	O	A-F-10 ,A-F-G12,A-E-15
LO12#	1	O	A-E-26 ,A-E-27 ,A-F-G28
LO13#	1	O	A-F-G36,A-E-37 ,A-E-38
LO14#	1	O	A-E-48 ,A-F-G44
LO15#	0	O	A-E-49 ,A-H-07 ,A-H-G04,A-U-22
LO16#	0	O	A-G-21 ,A-H-G20,A-J-23
LO17#	0	O	A-G-47 ,A-J-47 ,A-H-G44
LO18#	0	O	A-J-15 ,A-H-G12
LO19#	0	O	A-J-31 ,A-H-G28
LO20#	0	O	A-J-39 ,A-H-G36
LO21#	0	O	B-A-07 ,B-B-G04
LO22#	0	O	B-B-15 ,B-B-G12
LO23#	0	O	B-B-23 ,B-B-G20
LO24#	1	O	B-B-26 ,B-B-G28,B-B-31
LO25#	0	O	B-B-46 ,B-B-G44
LO26#	0	O	B-D-14 ,B-D-G12
LO27#	1	O	B-C-27 ,B-D-29 ,B-D-G28
LO28#	0	O	B-E-44 ,B-F-G44
LO29#	0	O	B-G-12 ,B-H-G12
LO30#	2	O	B-G-25 ,B-H-G28,B-H-30 ,B-H-31
LO31#	0	O	B-H-34 ,B-H-G36
LO32#	0	O	B-H-42 ,B-H-G44
LO33#	0	O	B-H-50 ,C-A-07 ,C-B-G04

NET	LD	DR	PINS
ROMLEVN-	1	10	B-B-05 ,B-G-39
ROMLODD-	1	10	B-B-14 ,B-F-24
ROMLOW-	2	10	B-A-15 ,B-B-11 ,B-B-04
RTS	0	10	C-F-19 ,A-U-18
RTS-	1	10	C-F-17 ,C-G-09
S	2	0	A-U-23 ,A-C-31 ,A-C-42
SDI	1	10	C-F-43 ,C-G-04
SDI-	1	0	C-F-41 ,A-U-43
SDO	1	10	C-E-17 ,C-G-14
SDO-	0	10	C-E-18 ,A-U-19
TBRE	0	10	C-C-07 ,C-G-07
UARTEN-	3	10	A-C-26 ,A-J-46 ,B-J-31 ,C-G-01
V+12	1	0	D-A-10 ,D-A-09 ,C-F-16 ,C-G-26
V-12	1	0	C-E-16 ,C-G-22
WR	0	10	A-U-10 ,B-U-36 ,B-H-49
WR-	9	20	A-J-18 ,A-C-12 ,A-D-40 ,B-H-48 ,C-A-35 ,C-B-30 ,C-F-29 ,B-J-17 ,B-J-04 ,C-E-02
WRITE-	1	10	A-U-33 ,B-C-21 ,C-B-32
XACK-	2	0	B-U-37 ,A-C-23 ,A-C-30
XIO0-	0	10	B-U-10 ,B-J-33 ,C-J-29
XIO1-	0	10	B-U-33 ,B-J-34 ,C-J-30
XIO2-	0	10	B-U-11 ,B-H-33 ,C-J-31
X1	1	0	A-B-31 ,A-A-44 ,A-A-45
X2	1	0	A-A-46 ,A-B-44 ,A-B-32
0WAIT	0	10	A-A-40 ,A-E-10
1WAIT	0	10	A-A-41 ,A-E-11
2WAIT	0	10	A-A-42 ,A-E-12
3WAIT	0	10	A-A-43 ,A-E-13

PRECEDING PAGE BLANK NOT FILMED

ADD	A-E-14	TO	B-J-49	CLK-
REMOVE	A-A-19	TO	A-J-20	CLK
REMOVE	A-A-19	TO	A-A-24	CLK
ADD	A-A-24	TO	A-J-20	CLK
ADD	A-A-19	TO	B-J-50	CK

DIPCON	2	$0.00
DIP14	6	$0.00
DIP16	2	$0.00
HC00	2	$0.00
HC02	1	$0.00
HC04	2	$0.00
HC08	1	$0.00
HC11	1	$0.00
HC123A	1	$0.00
HC138	1	$0.00
HC191	1	$0.00
HC194	1	$0.00
HC27	1	$0.00
HC30	2	$0.00
HC32	3	$0.00
HC4075	2	$0.00
HC4078	3	$0.00
HC74	2	$0.00
HI104	5	$0.00
HM6516	2	$0.00
LOW104	1	$0.00
MC1488	1	$0.00
MC1489	1	$0.00
O512DT	1	$0.00
27C64	4	$0.00
80C86	1	$0.00
82C52	1	$0.00
82C54	1	$0.00
82C59A	1	$0.00
82C82	3	$0.00
82C84	1	$0.00
82C86	4	$0.00
TOTAL	61	$0.00

*** DE-LANG CONTROLLER COMPUTER BOARD ***

OCT 10, 1986

CREM.UG1.V3 -- *WIREWRAP BOARD DESCRIPTION PACKAGE -- CBM & AD 20-DECEMBER-1979

BAY-A

A-A	A-B	A-C	A-D	A-E	A-F	A-G	A-H	A-J
01 LO03#		HO01# 01 I	01 LANGRDY	HO07# 01 I	01 AD8	HO13# 01 I	01 HO13#	HO13# 01 I
02 AD14		AD15 02 I	02 RDY2EN	02 I	02 AD9	D8 02 I	02 HO13#	02 I
03 AD13		ADR16 03 I	03 RDY2	03 I	03 AD10	D9 03 I	03 HO13#	A15 03 I
04 AD12		ADR17 04 I	04 MEM-/IO	04 I	04 AD11	D10 04 I	04 A19	A16 04 I
05 AD11		ADR18 05 I	05 INTA-	05 I	05 AD12	D11 05 I	05 A18	05 I
06 AD10		ADR19 06 I	06 IODEC	06 I	06 AD13	D12 06 I	06 A17	06 I
07 AD9		BHE 07 I	07 LO10#	07 I	07 AD14	D13 07 I	07 LO15#	ROMADR- 07 I
08 AD8		HO01# 08 I	----------	----------	08 AD15	D14 08 I		----------
09 AD7		RD- 09 I	09 ALE-	HO08# 09 I	09 DEN-	D15 09 I		09 RAMADR-
10 AD6		LO01# 10 I	10 HO09#	OWAIT 10 I	10 LO11#	DT/R- 10 I		10 MEM-/IO
11 AD5		11 I	11	1WAIT 11 I		----------		11 RAMACK-
12 AD4		WR- 12 I	12	2WAIT 12 I		12 AD0	HO14# 12 I	12 ROMADR-
13 AD3		MEM/IO- 13 I	13	3WAIT 13 I		13 AD1	D0 13 I	13 MEM-/IO
14 AD2		DT/R- 14 I	14	CLK- 14 I		14 AD2	D1 14 I	14 ROMACK-
15 AD1		DEN- 15 I	15	LO11# 15 I		15 AD3	D2 15 I	15 LO18#
16 AD0		ALE 16 I	16 LO01#	HO08# 16 I		16 AD4	D3 16 I	----------
17 NMI		INTA- 17 I		----------		17 AD5	D4 17 I	17 RD-
18 INTR		LO02# 18 I		18 ADR16	HO11# 18 I	18 AD6	D5 18 I	18 WR-
19 CK		READY 19 I		19 ADR17	A16 19 I	19 AD7	D6 19 I	19 MEM/IO-
20 LO04#		RESET 20 I		20 ADR18	A17 20 I	20 DEN-	D7 20 I	20 CLK
----------				21 ADR19	A18 21 I	21 LO16#	DT/R- 21 I	21 ILADCLR-
22 RD-	HO02# 22 I	22 HO04#	HO04# 22 I	22 HO11#	A19 22 I	----------		22 MEMCLK
23 RD	A3 23 I	23 XACK-	23 I	23 HO11#	23 I	23 AD7	HO15# 23 I	23 LO16#
24 CLK	A3- 24 I	24 PITEN-	ADRINT- 24 I	24 HO11#	24 I	24 AD6	D9 24 I	----------
25 CLK-	A14 25 I	25 PICEN-	ROMACK- 25 I	25 HO10#	25 I	25 AD5	D10 25 I	25 CYCSET-
26 ALE	A14- 26 I	26 UARTEN-	26 I	26 LO12#	26 I	26 AD4	D11 26 I	26 HO16#
27 ALE-	MEM/IO- 27 I	27 RAMACK-	27 I	27 LO12#	ALE 27 I	27 AD3	D12 27 I	27 ALE
28 LO05#	MEM-/IO 28 I	28 LO08#	RDY1 28 I	----------	----------	28 AD2	D13 28 I	28 HO16#
----------	----------	----------	----------	29 AD8	HO10# 29 I	29 AD1	D14 29 I	29 EOCEN-
30 LO05#	HO03# 30 I	30 XACK-	HO05# 30 I	30 AD9	A8 30 I	30 AD0	D15 30 I	30
31 PCLK	X1 31 I	31 S	AD8 31 I	31 AD10	A9 31 I	----------		31 LO19#
32 ALE	X2 32 I	32 R	AD9 32 I	32 AD11	A10 32 I	32 D8	HO16# 32 I	----------
33 RDY1	LO06# 33 I	33 ADR19	AD10 33 I	33 AD12	A11 33 I	33 D7	IR2 33 I	33 EOCEN-
34 READY	34 I	34 ADR18	AD11 34 I	34 AD13	A12 34 I	34 D6	IR3 34 I	34 ILADCLR-
35 RDY2	LO06# 35 I	35 ADR17	AD12 35 I	35 AD14	A13 35 I	35 D5	IR4 35 I	35 EOCSET-
36 ALE	36 I	36 ADR16	AD13 36 I	36 AD15	A14 36 I	36 D4	IR5 36 I	36 ROMACK-
37 CLK	RES- 37 I	37 AD15	AD14 37 I	37 LO13#	A15 37 I	37 D3	IR6 37 I	37 RD-
38 LO06#	RESET 38 I	----------	----------	38 LO13#	ALE 38 I	38 D2	IR7 38 I	38 ILLROM
----------	----------	39 LANGRD-	HO06# 39 I	----------	----------	39 D1	D0 39 I	39 LO20#
40 OWAIT	RDY2EN 40 I	40 LANGWR-	WR- 40 I	40 ADO	HO12# 40 I	----------		----------
41 1WAIT	NMI 41 I	41 LANGRDY	RD- 41 I	41 AD1	A0 41 I	41 RAMADR-	HO17# 41 I	41 MEMDIS
42 2WAIT	LO07# 42 I	42 S	ILADCLR- 42 I	42 AD2	A1 42 I	42 A19	42 I	42 MEMEN-
43 3WAIT	43 I	43 FEEDBACK	RES- 43 I	43 AD3	A2 43 I	43 A18	A12 43 I	43 ILADCLR-
44 X1	X2 44 I	44 RES-	R 44 I	44 AD4	A3 44 I	44 A17	A13 44 I	44 ILLUART
45 X1	LO07# 45 I	45 LO09#	FEEDBACK 45 I	45 AD5	A4 45 I	45 A16	A14 45 I	45 A3-
46 X2	LO07# 46 I			46 AD6	A5 46 I	46	A15 46 I	46 UARTEN-
				47 AD7	A6 47 I	47 LO17#	47 I	47 LO17#
				48 LO14#	A7 48 I			
				49 LO15#	ALE 49 I			
				50	50 I			

PRECEDING PAGE BLANK NOT FILMED

*** DE-LANG CONTROLLER COMPUTER BOARD *** OCT 10, 1986

CREM.UG1.V3 -- *WIREWRAP BOARD DESCRIPTION PACKAGE -- CBM & AD 20-DECEMBER-1979

BAY-C

C-A	C-B	C-C	C-D	C-E	C-F	C-G	C-H	C-J
01 A3-	HO38# 01 I	01	IR7 01 I	01 RD-		UARTEN- 01 I	01 DO	HO46# 01 I
02 A2-	PICEN- 02 I	02 ENDCONV	IR6 02 I	02 WR-		HO46# 02 I	02 D1	LANGDO 02 I
03	ILLPIC 03 I	03 ADRINT-	IR5 03 I	03 DO		DR 03 I	03 D2	LANGD1 03 I
04	04 I	04 PITOUTO	IR4 04 I	04 D1		SDI 04 I	04 D3	LANGD2 04 I
05	05 I	05 PITOUT1	IR3 05 I	05 D2		05 I	05 D4	LANGD3 05 I
06	06 I	06 PITOUT2	IR2 06 I	06 D3		RESET 06 I	06 D5	LANGD4 06 I
07 LO33#	07 I	07 TBRE	IR1 07 I	07 D4		TBRE 07 I	07 D6	LANGD5 07 I
		08 DR	IRO 08 I	08 D5		08 I	08 D7	LANGD6 08 I
09	HO39# 09 I			09 D6		RTS- 09 I	09 LANGDEN-	LANGD7 09 I
10 IOJUMPO	ILIOADR- 10 I	10 IR1	HO42# 10 I	10 D7		DTR- 10 I	10 LO43#	READ- 10 I
11 IOJUMP1	ILLPIC 11 I	11 IRO	LANGD6 11 I	11 A1		DSR- 11 I		
12 IOJUMP2	ILLPIT 12 I	12 GATEO	LANGD7 12 I	12 A2		CTS- 12 I		
13 ILLADRO	ILLUART 13 I	13 GATE1	LANGD8 13 I	13 IX		LO43# 13 I		
14	ILLADR1 14 I	14 GATE2	LANGD9 14 I	14 OX		SDO 14 I		
15 LO34#	15 I	15 NOLANGO	LANGD10 15 I					
		16 NOLANG1	LANGD11 16 I	16 V-12	V+12 16 I	16 HO47#		LO43# 16 I
17 IOJUMPO	LO35# 17 I	17 NOLANG2	NOLANG3 17 I	17 SDO	RTS- 17 I	17		17 I
18 IOJUMP1	18 I			18 SDO-	HO48# 18 I	18		18 I
19 IOJUMP2	19 I	19 LO38#	HO43# 19 I	19 DTR-	RTS 19 I	19		19 I
20	20 I	20 ADRTRIG	REXT 20 I	20 HO48#	20 I	20		20 I
21	21 I	21 HO43#	CEXT 21 I	21 DTR	HO48# 21 I	21		21 I
22	22 I	22 ADRINT-	22 I	22 GND12	22 I	22 V-12		22 I
23	23 I	23	23 I			23		23 I
24	24 I	24	24 I			24 GND12		24 I
		25	25 I			25		25 I
26 ILLIOEN-	HO40# 26 I	26 LO39#	26 I			26 V+12		26 I
27 ILIOADR-	27 I					27		27 I
28 ILLIO-	28 I		28 D7		HO44# 28 I			
29 LANGRD-	29 I		29 D6		WR- 29 I	29 D8	HO50# 29 I	29 XIOO-
30 RD-	WR- 30 I		30 D5		RD- 30 I	30 D9	LANGD8 30 I	30 XIO1-
31 READ-	LANGWR- 31 I		31 D4		PITEN- 31 I	31 D10	LANGD9 31 I	31 XIO2-
32 LO36#	WRITE- 32 I		32 D3		A2 32 I	32 D11	LANGD10 32 I	32 PITCKO
			33 D2		A1 33 I	33 D12	LANGD11 33 I	33 PITCK1
34 PICEN-		HO41# 34 I	34 D1		CLK2 34 I	34 D13	NOLANG3 34 I	34 PITCK2
35 WR-		A1 35 I	35 DO		PITOUT2 35 I	35 D14	NOLANG2 35 I	35 PITCK3
36 RD-		INTA- 36 I	36 CLKO		GATE2 36 I	36 D15	NOLANG1 36 I	
37 D7		IR7 37 I	37 PITOUTO		CLK1 37 I	37 LANGDEN-	NOLANGO 37 I	
38 D6		IR6 38 I	38 GATEO		GATE1 38 I	38 LO44#	READ- 38 I	
39 D5		IR5 39 I	39 LO4O#		PITOUT1 39 I			
40 D4		IR4 40 I					40 LANGD5	HO51# 40 I
41 D3		IR3 41 I	41 LO41#	HO45# 41 I	41 SDI-	HO49# 41 I	41 LANGD4	41 I
42 D2		IR2 42 I	42 PITCK1	LO42# 42 I	42	42 I	42 LANGD3	42 I
43 D1		IR1 43 I	43 PITCKO	PCLK 43 I	43 SDI	43 I	43 LANGD2	43 I
44 DO		IRO 44 I	44 LO41#	44 I	44 DSR	44 I	44 LANGD1	44 I
45		INTR 45 I	45 LO41#	45 I	45	CTS 45 I	45 LANGDO	45 I
46		46 I	46 PITCK2	RES- 46 I	46 DSR-	46 I		46 I
47 LO37#		47 I	47 PITCK3	LO42# 47 I	47 LO42#	CTS- 47 I		47 I
			48 LO41#	LO42# 48 I				

PRECEDING PAGE BLANK NOT FILMED

*** DE-LANG CONTROLLER COMPUTER BOARD ***

OCT 10, 1986

CREM.UG1.V3 -- *WIREWRAP BOARD DESCRIPTION PACKAGE -- CBM & AD 20-DECEMBER-1979

I/O CONNECTORS

A-U-

01	LANGD10	LANGD11	24
02	LANGD8	LANGD9	25
03	LANGD6	LANGD7	26
04	LANGD4	LANGD5	27
05	LANGD2	LANGD3	28
06	LANGDO	LANGD1	29
07	A2	A3	30
08			31
09	READ-	A1	32
10	WR	WRITE-	33
11	LO10#	ENDCONV	34
12	LO10#	LO10#	35
13	LO46#	LO46#	36
14		LO46#	37
15			38
16			39
17	DTR	GND12	40
18	RTS	DSR	41
19	SDO-	CTS	42
20		SDI-	43
21			44
22	LO15#		45
23	S	R	46

B-U-

01	D15	D14	24
02	D13	D12	25
03	D11	D10	26
04	D9	D8	27
05	D7	D6	28
06	D5	D4	29
07	D3	D2	30
08	D1	DO	31
09			32
10	XIOO-	XIO1-	33
11	XIO2-	A1	34
12	AO	A3	35
13	A2	WR	36
14	RD	XACK-	37
15			38
16			39
17			40
18			41
19			42
20			43
21			44
22			45
23			46

C-U-

01	24
02	25
03	26
04	27
05	28
06	29
07	30
08	31
09	32
10	33
11	34
12	35
13	36
14	37
15	38
16	39
17	40
18	41
19	42
20	43
21	44
22	45
23	46

D-U-

01	24
02	25
03	26
04	27
05	28
06	29
07	30
08	31
09	32
10	33
11	34
12	35
13	36
14	37
15	38
16	39
17	40
18	41
19	42
20	43
21	44
22	45
23	46

*** DE-LANG CONTROLLER COMPUTER BOARD ***

OCT 10, 1986

CREM.UG1.V3 -- *WIREWRAP BOARD DESCRIPTION PACKAGE -- CBM & AD 20-DECEMBER-1979

PLUGGING LIST

PLUG PIN 1 OF:	INTO:
80C86	A-A-01
HC04	A-A-22
82C84	A-A-30
DIP14	A-A-40
HC30	A-C-22
HI104	A-C-30
HC00	A-C-39
HC08	A-D-01
HC194	A-D-09
82C82	A-E-18
82C82	A-E-29
82C82	A-E-40
82C86	A-F-01
82C86	A-G-12
HI104	A-G-23
HI104	A-G-32
HC4078	A-G-41
HC30	A-H-01
HC32	A-J-09
HC11	A-J-17
HC74	A-J-25
HC32	A-J-33
HC02	A-J-41
HC4075	B-A-01
HC4075	B-B-09
HC00	B-B-17
HC74	B-B-25
27C64	B-B-33
27C64	B-D-01
27C64	B-D-16
27C64	B-E-31
HM6516	B-G-01
HM6516	B-G-14
HC138	B-H-27
HC4078	B-H-36
HC04	B-H-44
HC27	C-A-01
HC4078	C-A-09
LOW104	C-A-17
HC32	C-A-26
82C59A	C-A-34
DIP16	C-C-01

Appendix B

Software Source Listings

1. MON86
2. VRTX and TRACER Board Support Packages

```
                        1   ;
                        2   ;*************************************************************************
                        3   ;*                                                                       *
                        4   ;*                          8086 Mini-monitor                            *
                        5   ;*                                                                       *
                        6   ;*                    by Steve Rowe and Dan Beckett                      *
                        7   ;*                                                                       *
                        8   ;*                              27-Mar-86                                *
                        9   ;*                                                                       *
                       10   ;*  Modified for Langmuir probe by S. Rowe and J. Seldin Aug/Sep '86      *
                       11   ;*  Modified to allow 16-bit I/O Oct. '86                                 *
                       12   ;*                                                                       *
                       13   ;*************************************************************************
                       14   ;
                       15   ;
                       16   ;       8252 UART Status Register and Input/Output Register Addresses
                       17   ;
= 0032                 18   USRADR  EQU     032H            ; UART Status Register Address
= 0036                 19   MSRADR  EQU     036H            ; Modem Status Register Address
= 0030                 20   URTIN   EQU     030H            ; Data from UART
= 0030                 21   URTOUT  EQU     030H            ; Data to UART
                       22   ;
                       23   ;       8259A PIC Control Word Address
                       24   ;
= 0020                 25   ICW1AD  EQU     020H            ; Address to store ICW1
= 0022                 26   ICW2AD  EQU     022H            ; Address to store ICW2
                       27   ;
= 0016                 28   NREGS   EQU     22              ; number of register names
                       29   ;
                       30   ;
```

```
                                        50              PAGE
                                        51      ;
0000                                    52      MONSEG  SEGMENT                         ; ROMable code follows
                                        53              ASSUME  CS:MONSEG, DS:MONDAT
                                        54      ;
                                        55      ;       ROMable data area (text messages, etc)
                                        56      ;
0000  07 0D 0A                          57      ERRMSG  DB      07,13,10        ; Syntax error message text
0003  49 6E 76 61 6C 69                 58              DB      'Invalid '
      64 20                             58
000B  6D 6F 6E 69 74 6F                 59              DB      'monitor '
      72 20                             59
0013  63 6F 6D 6D 61 6E                 60              DB      'command!',0
      64 21 00                          60
                                      * 61      ;
001C  0D 0A 4D 4F 4E 38                 62      PROMPT  DB      13,10,'MON86>',0 ; Prompt string
      36 3E 00                          62
                                        63      ;
0025  08 20 08 00                       64      BCKSPC  DB      08,32,08,0      ; Backspace sequence
0029  1B 5B 48 1B 5B 4A                 65      CLRSCR  DB      01BH,'[','H',01BH,'[','J',0     ; Terminal CLS
      00                                65
                                        66      ;
0030  4158 4258 4358                    67      REGNMS  DW      'AX','BX','CX'  ; Register names
0036  4458 5350 4250                    68              DW      'DX','SP','BP'
003C  5349 4449 4353                    69              DW      'SI','DI','CS'
0042  4453 5353 4553                    70              DW      'DS','SS','ES'
0048  4950 464C 414C                    71              DW      'IP','FL','AL'
004E  4148 424C 4248                    72              DW      'AH','BL','BH'
0054  434C 4348 444C                    73              DW      'CL','CH','DL'
005A  4448                              74              DW      'DH'
                                        75      ;
005C  0D 0A 3F 2D 4D 4F                 76      PARMSG  DB      13,10,'?-MON86- Invalid Parameter'
      4E 38 36 2D 20 49                 76
      6E 76 61 6C 69 64                 76
      20 50 61 72 61 6D                 76
      65 74 65 72                       76
0078  07 00                             77              DB      07,0
                                        78      ;
007A  30 31 32 33 34 35                 79      HEXTBL  DB      '0','1','2','3','4','5','6','7'   ; Used in BINASC
      36 37                             79
0082  38 39 41 42 43 44                 80              DB      '8','9','A','B','C','D','E','F'
      45 46                             80
                                        81      ;
```

```
                                96              PAGE
                                97      ;
                                98      ;       Set up data and stack segment
                                99      ;
                               100              PUBLIC  MONIT
019C  FA                       101      MONIT:  CLI                     ; disable interrupts
019D  1E                       102              PUSH    DS
019E  50                       103              PUSH    AX
019F  B0 FF                    104              MOV     AL,0FFH         ; Mask all inputs to PIC
01A1  E6 22                    105              OUT     ICW2AD,AL
01A3  B8  ---- R               106              MOV     AX,MONDAT       ; get segment address of system RAM
01A6  8E D8                    107              MOV     DS,AX           ; store it in the data segment register
01A8  58                       108              POP     AX
01A9  A3 0050 R                109              MOV     REGS,AX
01AC  89 1E 0052 R             110              MOV     REGS+2,BX
01B0  89 0E 0054 R             111              MOV     REGS+4,CX
01B4  89 16 0056 R             112              MOV     REGS+6,DX
01B8  8B C4                    113              MOV     AX,SP
01BA  05 0008                  114              ADD     AX,8            ; Add offset of Interrupt to stack ptr
01BD  A3 0058 R                115              MOV     REGS+8,AX
01C0  89 2E 005A R             116              MOV     REGS+10,BP
01C4  89 36 005C R             117              MOV     REGS+12,SI
01C8  89 3E 005E R             118              MOV     REGS+14,DI
01CC  58                       119              POP     AX              ; Get DS into AX
01CD  5B                       120              POP     BX              ; Get old Inst. Ptr into BX
01CE  59                       121              POP     CX              ; Get old CS into CX
01CF  89 0E 0060 R             122              MOV     REGS+16,CX      ; save CS
01D3  A3 0062 R                123              MOV     REGS+18,AX      ; save DS
01D6  8C D1                    124              MOV     CX,SS
01D8  89 0E 0064 R             125              MOV     REGS+20,CX      ; save SS
01DC  8C C0                    126              MOV     AX,ES
01DE  A3 0066 R                127              MOV     REGS+22,AX      ; save ES
01E1  89 1E 0068 R             128              MOV     REGS+24,BX      ; save old IP
01E5  58                       129              POP     AX              ; get old Flags from stack
01E6  A3 006A R                130              MOV     REGS+26,AX      ; save old PSW
```

INTO PORT #20
00 " " #22
" " #24

```
                           PAGE
                 ;
                 ;         Now figure out which command was asked for
                 ;
0219  80 3E 006C R 44          CMP     CMD,'D' ; display command?
021E  75 08                    JNE     LL1
0220  BA 0010                  MOV     DX,16
0223  E8 02DF R                CALL    DSPLAY          ; yes, call routine
0226  EB D0                    JMP     MLOOP
                 ;
0228  80 3E 006C R 49    LL1:  CMP     CMD,'I' ; input from port command?
022D  75 05                    JNE     LL2
022F  E8 0375 R                CALL    INPUT           ; yes, call routine
0232  EB C4                    JMP     MLOOP
                 ;
0234  80 3E 006C R 4F    LL2:  CMP     CMD,'O' ; output to port command?
0239  75 05                    JNE     LL3
023B  E8 02D6 R                CALL    OUTPUT          ; yes, call routine
023E  EB B8                    JMP     MLOOP
                 ;
0240  80 3E 006C R 52    LL3:  CMP     CMD,'R' ; register command?
0245  75 05                    JNE     LL4
0247  E8 0456 R                CALL    REGSTR          ; yes, call routine
024A  EB AC                    JMP     MLOOP
                 ;
024C  80 3E 006C R 53    LL4:  CMP     CMD,'S' ; set memory command?
0251  75 05                    JNE     LL5
0253  E8 03D6 R                CALL    STORE           ; yes, call routine
0256  EB A0                    JMP     MLOOP
                 ;
0258  80 3E 006C R 20    LL5:  CMP     CMD,' ' ; no command?
025D  74 99                    JE      MLOOP           ; yes, get new input line
025F  80 3E 006C R 51          CMP     CMD,'Q' ; quit command?
0264  75 45                    JNE     LL6
                                                         mask all
0266  B0 FF                    MOV     AL,0FFH         ; enable only monitor interrupts
0268  E6 22                    OUT     ICW2AD,AL
026A  B0 20                    MOV     AL,020H         ; Get end of interrupt command
026C  E6 20                    OUT     ICW1AD,AL       ; tell PIC to receive
026E  8B 26 0058 R             MOV     SP,REGS+8       ; restore stack pointer
0272  A1 0064 R                MOV     AX,REGS+20      ; and stack segment register.
0275  8E D0                    MOV     SS,AX
0277  A1 006A R                MOV     AX,REGS+26      ; Get flags register
027A  50                       PUSH    AX              ; Put it on the stack for IRET
027B  A1 0060 R                MOV     AX,REGS+16      ; Get old code segment
027E  50                       PUSH    AX              ; Put it on the stack for IRET
027F  A1 0068 R                MOV     AX,REGS+24      ; Get instruction pointer
0282  50                       PUSH    AX              ; IRET stack is ready.  Now restore others
0283  A1 0066 R                MOV     AX,REGS+22
0286  8E C0                    MOV     ES,AX
0288  8B 3E 005E R             MOV     DI,REGS+14
028C  8B 36 005C R             MOV     SI,REGS+12
0290  8B 2E 005A R             MOV     BP,REGS+10
0294  8B 16 0056 R             MOV     DX,REGS+6
0298  8B 0E 0054 R             MOV     CX,REGS+4
```

```
                        PAGE
                        ;
                        ;*************************************************************************
                        ;*
                        ;*      OUTPUT to I/O port
                        ;*
                        ;*************************************************************************
                        ;
02D6  8B 16 006F R      OUTPUT: MOV     DX,PAR2         ; get port number
02DA  A1 006D R                 MOV     AX,PAR1         ; get number to output
                        ;
02DD  EF                        OUT     DX,AX           ; output value
                        ;
02DE  C3                        RET                     ; return
                        ;
```

```
                                282             PAGE
030E  8A C6                     283             MOV     AL,DH           ; get MSB of address
0310  E8 05E3 R                 284             CALL    BINASC          ; convert to 2 hex chars in BX
0313  8B C3                     285             MOV     AX,BX           ; get them in a convenient spot
0315  88 26 0002 R              286             MOV     CHRBUF+2,AH     ; store 'addr:' in output buffer
0319  A2 0003 R                 287             MOV     CHRBUF+3,AL
031C  8A C2                     288             MOV     AL,DL           ; get LSB of addr
031E  E8 05E3 R                 289             CALL    BINASC
0321  8B C3                     290             MOV     AX,BX
0323  88 26 0004 R              291             MOV     CHRBUF+4,AH
0327  A2 0005 R                 292             MOV     CHRBUF+5,AL
032A  C6 06 0006 R 3A           293             MOV     CHRBUF+6,':'
032F  C6 06 0007 R 20           294             MOV     CHRBUF+7,' '
0334  A0 006D R                 295             MOV     AL,BYTE PTR PAR1                ; get value
0337  E8 05E3 R                 296             CALL    BINASC          ; turn into ASCII
033A  88 3E 0008 R              297             MOV     CHRBUF+8,BH
033E  88 1E 0009 R              298             MOV     CHRBUF+9,BL
0342  C6 06 000A R 20           299             MOV     CHRBUF+10,' '
0347  C6 06 000B R 20           300             MOV     CHRBUF+11,' '
034C  C6 06 000C R 2E           301             MOV     CHRBUF+12,'.'
0351  3C 20                     302             CMP     AL,' '
0353  7C 03                     303             JL      DLL3
0355  A2 000C R                 304             MOV     CHRBUF+12,AL
0358  C6 06 000D R 00           305   DLL3:     MOV     CHRBUF+13,0
035D  8D 06 0000 R              306             LEA     AX,CHRBUF
0361  9A 0561 ---- R            307             CALL    FAR PTR SNDBUF  ; display line
0366  FF 06 0071 R              308             INC     ADDREG          ; point at next location
036A  59                        309             POP     CX
036B  49                        310             DEC     CX              ; too far to use LOOP
036C  83 F9 00                  311             CMP     CX,0
036F  74 03                     312             JE      DLL4
0371  E9 02EC R                 313             JMP     DLL2            ; loop until all addresses are shown
0374  C3                        314   DLL4:     RET                     ; return
```

```
                                358             PAGE
                                359     ;
                                360     ;*****************************************************************************
                                361     ;
                                362     ;       STORE in memory
                                363     ;
                                364     ;*****************************************************************************
                                365     ;
03D6   80 F9 01                 366     STORE:  CMP     CL,1
03D9   75 0C                    367             JNE     TOPARS          ; 2 pars
03DB   A1 006D R                368             MOV     AX,PAR1         ; copy value into second par
03DE   A3 006F R                369             MOV     PAR2,AX
03E1   A1 0071 R                370             MOV     AX,ADDREG       ; store address reg as first par
03E4   A3 006D R                371             MOV     PAR1,AX
                                372     ;
03E7   A1 006F R                373     TOPARS: MOV     AX,PAR2         ; get number to store
03EA   8B 2E 006D R             374             MOV     BP,PAR1         ; get address to store
03EE   8B 1E 0062 R             375             MOV     BX,REGS+18      ; Get simulated DS
03F2   1E                       376             PUSH    DS
03F3   8E DB                    377             MOV     DS,BX
03F5   3E: 88 46 00             378             MOV     DS:[BP],AL      ; store number
                                379     ;
03F9   1F                       380             POP     DS
03FA   BA 0001                  381             MOV     DX,1
03FD   E9 02DF R                382             JMP     DSPLAY          ; go display and increment
                                383     ;
```

```
                                426             PAGE
0436  B1 01                     427   ONEPAR:   MOV     CL,01           ; one parameter
0438  EB 1B 90                  428             JMP     DONE8           ; leave routine
                                429   ;
043B  B1 00                     430   ZERPAR:   MOV     CL,00           ; zero parameters
043D  EB 16 90                  431             JMP     DONE8           ; leave routine
                                432   ;
0440  8D 06 005C R              433   BADPAR:   LEA     AX,PARMSG       ; point at error message
0444  1E                        434             PUSH    DS
0445  8C CB                     435             MOV     BX,CS
0447  8E DB                     436             MOV     DS,BX
0449  9A 0561 ---- R            437             CALL    FAR PTR SNDBUF  ; print it
044E  1F                        438             POP     DS
044F  B1 FF                     439             MOV     CL,0FFH         ; error flag
0451  88 0E 0073 R              440             MOV     ERRFLG,CL       ; Store it as well
                                441   ;
0455  C3                        442   DONE8:    RET                     ; go back to caller
```

```
                                        PAGE
                                ;
                                ;*******************************************************************************
                                ;
                                ;       LGLPAR takes a pointer [BP+SI] to a parameter in a character
                                ;       buffer and returns:
                                ;         CL = 01, AL = register_index if param is a register name
                                ;         CL = 02, AX = param_value  if param is a valid hex number
                                ;         CL = 03, AX = ?  if param is none of the above
                                ;
                                ;*******************************************************************************
                                ;
0494  52                       LGLPAR: PUSH    DX              ; save workspace registers
0495  53                               PUSH    BX
                                ;
                                ;       get first two chars into AX
                                ;
0496  8B 84 0000 R                     MOV     AX,WORD PTR CHRBUF[SI]
049A  86 C4                            XCHG    AL,AH
049C  46                               INC     SI
049D  46                               INC     SI
                                ;
049E  9A 05A0 ---- R                   CALL    FAR PTR ASCBIN          ; try to convert to number
04A3  80 FF 00                         CMP     BH,00           ; is it an error?
04A6  75 22                            JNE     NOTNUM
                                ;
                                ;       so far so good, check to see if it's a two byte number
                                ;
04A8  80 BC 0000 R 20                  CMP     CHRBUF[SI],' '
04AD  74 44                            JE      DONE2
                                ;
                                ;       possibly a two byte number
                                ;
04AF  8B D3                            MOV     DX,BX           ; Save BX
04B1  8B 84 0000 R                     MOV     AX,WORD PTR CHRBUF[SI]  ; get next two chars
04B5  86 C4                            XCHG    AL,AH           ; put them in proper order
04B7  46                               INC     SI
04B8  46                               INC     SI
                                ;
04B9  9A 05A0 ---- R                   CALL    FAR PTR ASCBIN          ; attempt conversion
04BE  80 FF 00                         CMP     BH,00           ; error
04C1  75 20                            JNE     BADPR           ; yes, return
                                ;
                                ;       legal 4-digit hex number.  scoot bytes around
                                ;
04C3  8A E2                            MOV     AH,DL           ; Transefer both halves of number
04C5  8A C3                            MOV     AL,BL           ; into return parameter and
04C7  EB 2C 90                         JMP     DONE4           ; go home with legal value
                                ;
```

```
                                578             PAGE
                                579     ;
                                580     ;*****************************************************************************
                                581     ;
                                582     ;       GETLIN inputs a line from the keyboard and stores it in
                                583     ;       the buffer CHRBUF
                                584     ;
                                585     ;*****************************************************************************
                                586     ;
04FA  BE 004F                   587     GETLIN: MOV     SI,79           ; fill buffer with blanks
04FD  8D 2E 0000 R              588             LEA     BP,CHRBUF
0501  B0 20                     589             MOV     AL,' '
                                590     ;
0503  3E: 88 02                 591     BLOOP:  MOV     DS:[BP+SI],AL ; blank character
0506  4E                        592             DEC     SI              ; move to next character
0507  7D FA                     593             JGE     BLOOP           ; continue looping for all
                                594     ;
0509  BE 0000                   595             MOV     SI,0            ; point at first buffer position
                                596     ;
050C  E4 32                     597     CHWAIT: IN      AL,USRADR       ; Character in yet?
050E  A8 40                     598             TEST    AL,040H
0510  74 0A                     599             JZ      CONT
0512  1E                        600             PUSH    DS
0513  BB  ---- R                601             MOV     BX,MONDAT
0516  8E DB                     602             MOV     DS,BX
0518  A2 0074 R                 603             MOV     DS:USR,AL
051B  1F                        604             POP     DS
051C  24 80                     605     CONT:   AND     AL,80H          ; Mask off irrelevent bits
051E  3C 00                     606             CMP     AL,0
0520  74 EA                     607             JE      CHWAIT          ; Not ready yet. keep looping.
0522  E4 30                     608             IN      AL,URTIN        ; Get character
0524  3C 0D                     609             CMP     AL,13           ; carriage return pressed?
0526  74 38                     610             JE      GOTLIN          ; yes, exit routine
0528  3C 7F                     611             CMP     AL,07FH         ; a DEL character?
052A  74 04                     612             JE      CHRDEL          ; Yes.  Pretend it's a backspace
052C  3C 08                     613             CMP     AL,08           ; Backspace?
052E  75 1F                     614             JNE     NOCTRL          ; No.  It must be a normal char.
0530  83 FE 00                  615     CHRDEL: CMP     SI,0            ; If nothing to delete, don't.
0533  74 D7                     616             JE      CHWAIT
0535  55                        617             PUSH    BP              ; Else save regs and display bs.
0536  56                        618             PUSH    SI
0537  1E                        619             PUSH    DS
0538  8C C8                     620             MOV     AX,CS
053A  8E D8                     621             MOV     DS,AX
053C  8D 06 0025 R              622             LEA     AX,BCKSPC       ; send backspace sequence to term.
0540  9A 0561 ---- R            623             CALL    FAR PTR SNDBUF
0545  1F                        624             POP     DS
0546  5E                        625             POP     SI
0547  5D                        626             POP     BP
0548  4E                        627             DEC     SI              ; Delete character from buffer, too.
0549  3E: C6 02 20              628             MOV     BYTE PTR DS:[BP+SI],' ' ; put a blank back
054D  EB BD                     629             JMP     CHWAIT
                                630     ;
054F  3C 61                     631     NOCTRL: CMP     AL,061H         ; Is it lower-case?
```

```
                                642             PAGE
                                643     ;
                                644     ;*******************************************************************************
                                645     ;
                                646     ;       SNDBUF sends the buffer pointed to by DS:AX.  The string must be
                                647     ;       NUL-terminated.
                                648     ;
                                649     ;*******************************************************************************
                                650     ;
                                651     PUBLIC  SNDBUF
0561                            652     SNDBUF  PROC    FAR
0561  8B E8                    653             MOV     BP,AX                   ; get pointer to buffer
0563  BE 0000                  654             MOV     SI,0                    ; point to beginning
0566  3E: 8A 22                655     SNDLP:  MOV     AH,DS:[BP+SI]           ; get next character to send
0569  46                      656             INC     SI                      ; point at next character in list
056A  80 FC 00                657             CMP     AH,0                    ; If NUL, we're finished.
056D  74 24                   658             JE      SNDOVR                  ; leave when finished.
056F  83 FE 01                659     SNDWT:  CMP     SI,1
0572  75 11                   660             JNE     CONT1
0574  1E                      661             PUSH    DS
0575  BB  ---- R              662             MOV     BX,MONDAT
0578  8E DB                   663             MOV     DS,BX
057A  A0 0074 R               664             MOV     AL,DS:USR
057D  1F                      665             POP     DS
057E  3C 00                   666             CMP     AL,0
0580  74 03                   667             JE      CONT1
0582  EB 03 90                668             JMP     CONT2
0585  E4 32                   669     CONT1:  IN      AL,USRADR               ; ready to send next char?
0587  24 40                   670     CONT2:  AND     AL,40H                  ; mask off irrelevent bits
0589  3C 00                   671             CMP     AL,0                    ; ready for next char?
058B  74 E2                   672             JE      SNDWT                   ; NO. Wait some more
058D  8A C4                   673             MOV     AL,AH                   ; get character to send
058F  E6 30                   674             OUT     URTOUT,AL               ; and send it off.
0591  EB D3                   675             JMP     SNDLP
                              676     ;
0593  1E                      677     SNDOVR: PUSH    DS
0594  BB  ---- R              678             MOV     BX,MONDAT
0597  8E DB                   679             MOV     DS,BX
0599  C6 06 0074 R 00         680             MOV     DS:USR,0
059E  1F                      681             POP     DS
059F  CB                      682             RET                             ; Back to caller
05A0                          683     SNDBUF  ENDP
```

```
05DB  D0 E3          738          SHL     BL,1
05DD  0A D8          739          OR      BL,AL          ; blend in LSB bits
05DF  B7 00          740          MOV     BH,0           ; clear error flag
05E1  EB EE          741          JMP     LEEVE          ; go home
                     742  ;
```

```
                                PAGE
                ;*****************************************************************
                ;
                ;       DISREG displays the value of the register whose name is
                ;       pointed at by REGNMS[SI]
                ;
                ;*****************************************************************
                ;
0602  83 FE 2C  DISREG: CMP     SI,NREGS*2      ; Is it a valid register?
0605  7F 57             JG      BDPAR           ; No.  display message & return home
0607  2E: 8B 84 0030 R  MOV     AX,CS:REGNMS[SI]        ; Get reg. name
060C  C6 06 0000 R 0D   MOV     CHRBUF,13       ; carriage return and
0611  C6 06 0001 R 0A   MOV     CHRBUF+1,10     ; line feed at beginning
0616  88 26 0002 R      MOV     CHRBUF+2,AH     ; Put 1st char. in output buffer
061A  A2 0003 R         MOV     CHRBUF+3,AL     ; Put 2nd char. in output buffer
061D  C6 06 0004 R 3A   MOV     CHRBUF+4,':'    ; Format output nicely
0622  C6 06 0005 R 20   MOV     CHRBUF+5,' '
                ;
0627  83 FE 1C          CMP     SI,28           ; Past 14th register?
062A  7D 44             JNL     BIT8            ; YES.  Must be an 8-bit reg.
                ;
```

```
                                808             PAGE
                                809     ;
                                810     ;       Illegal register.  Print error message
                                811     ;
065E  8D 06 005C R              812     BDPAR:  LEA     AX,PARMSG       ; point at error message
0662  1E                        813             PUSH    DS
0663  8C CB                    814             MOV     BX,CS
0665  8E DB                    815             MOV     DS,BX
0667  9A 0561 ---- R            816             CALL    SNDBUF          ; print it
066C  1F                        817             POP     DS
066D  EB 2A 90                  818             JMP     LEEV
                                819     ;
                                820     ;       8-bit register handler
                                821     ;
0670  8B C6                     822     BIT8:   MOV     AX,SI           ; Put SI into a math register
0672  2D 001C                   823             SUB     AX,01CH         ; Trim off fat
0675  D1 E8                     824             SHR     AX,1            ; Divide by 2 to point at correct entry
0677  8B F0                     825             MOV     SI,AX           ; Restore SI
0679  8D 2E 0050 R              826             LEA     BP,REGS         ; Point at table of register values
067D  3E: 8A 02                 827             MOV     AL,DS:[BP+SI]   ; Get value of register from table
                                828     ;
0680  E8 05E3 R                 829             CALL    BINASC          ; convert value to ASCII coded HEX
0683  88 3E 0006 R              830             MOV     CHRBUF+6,BH     ; move digits into output buffer
0687  88 1E 0007 R              831             MOV     CHRBUF+7,BL
068B  C6 06 0008 R 00           832             MOV     CHRBUF+8,0      ; NULL char. line terminator
                                833     ;
0690  8D 06 0000 R              834             LEA     AX,CHRBUF       ; Send line to terminal
0694  9A 0561 ---- R            835             CALL    SNDBUF
                                836     ;
                                837     ;       Restore registers and leave
                                838     ;
0699  C3                        839     LEEV:   RET
069A                            840     MONSEG  ENDS
                                841             END                     ; OF ASSEMBLY
```

									FS	roSE	F	-286	.17	7-0(	86	:20	mbo	

LL2.	L NEAR	0234	MONSEG	
LL3.	L NEAR	0240	MONSEG	
LL4.	L NEAR	024C	MONSEG	
LL5.	L NEAR	0258	MONSEG	
LL6.	L NEAR	02AB	MONSEG	
LL7.	L NEAR	02C4	MONSEG	
LOOP6.	L NEAR	04D3	MONSEG	
LOOPR.	L NEAR	0488	MONSEG	
LSBCNV	L NEAR	05BB	MONSEG	
MLOOP.	L NEAR	01F8	MONSEG	
MONIT.	L NEAR	019C	MONSEG	Global
MSRADR	Number	0036		
NOCTRL	L NEAR	054F	MONSEG	
NOPARS	L NEAR	0482	MONSEG	
NOTLWR	L NEAR	0555	MONSEG	
NOTNUM	L NEAR	04CA	MONSEG	
NREGS.	Number	0016		
NUMOK.	L NEAR	05D3	MONSEG	
ONEPAR	L NEAR	0436	MONSEG	
OUTPUT	L NEAR	02D6	MONSEG	
PAR1	L WORD	006D	MONDAT	
PAR2	L WORD	006F	MONDAT	
PARMSG	L BYTE	005C	MONSEG	Length =001C
PARSE.	L NEAR	0400	MONSEG	
PROMPT	L BYTE	001C	MONSEG	Length =0009
REGNMS	L WORD	0030	MONSEG	Length =0003
REGS	L WORD	0050	MONDAT	Length =000E
REGSTO	L NEAR	046D	MONSEG	
REGSTR	L NEAR	0456	MONSEG	
SNDBUF	F PROC	0561	MONSEG	Global Length =003F
SNDLP.	L NEAR	0566	MONSEG	
SNDOVR	L NEAR	0593	MONSEG	
SNDWT.	L NEAR	056F	MONSEG	
STORE.	L NEAR	03D6	MONSEG	
TOPARS	L NEAR	03E7	MONSEG	
URTIN.	Number	0030		
URTOUT	Number	0030		
USR.	L BYTE	0074	MONDAT	
USRADR	Number	0032		
ZERPAR	L NEAR	043B	MONSEG	

Warning Errors	Severe Errors
0	0

```
1   ;*****************************************************************************
2   ;                                                                            *
3   ;                    Board support package for the Seldin-1                  *
4   ;                  Implementing VRTX and TRACER  (hopefully)                 *
5   ;                       With polled downloading routine                      *
6   ;                          and XON/XOFF transmission                         *
7   ;                               Version 0.2                                  *
8   ;                                                                            *
9   ;       Programmer..................... Eugene Walden                        *
10  ;       Supervisor..................... Bruce Block                          *
11  ;       Date........................... 5-1-87                               *
12  ;                                                                            *
13  ;*****************************************************************************
14
15  ;*****************************************************************************
16  ;                                                                            *
17  ;                              SYSTEM CONFIGURATION                          *
18  ;                              --------------------                          *
19  ;     This system will implement VRTX and TRACER using a two- vector system. *
20  ; the way in which VRTX is called in version 0.0 is with an interrupt. If    *
21  ; TRACER is added into the system, four possible setups present themselves:  *
22  ;       1) single vector, single I/O channel                                 *
23  ;       2) single vector, two I/O channels                                   *
24  ;       3) two vectors, single I/O channel                                   *
25  ;       4) two vectors, two I/O channels                                     *
26  ; I will use method #2.                                                      *
27  ;     When adding TRACER to an existing board support package with VRTX, the *
28  ; following additions must be made:                                          *
29  ;       1) add interrupt # 1 for trace mode                                  *
30  ;       2) add interrupt # 3 for support of breakpoints                      *
31  ;       3) add an interrupt in the IVT somewhere which has the starting      *
32  ;          point of TRACER if the two vector method is being used.           *
33  ;       4) construct the configuration table for TRACER                      *
34  ;       5) place the address of the configuration table for TRACER at an off-*
35  ;          set of -4 from the beginning of the configuration table for VRTX. *
36  ;       6) call TR_INIT before VRTX_INIT or VRTX_GO                          *
37  ;                                                                            *
38  ;*****************************************************************************
39
40  ;*****************************************************************************
41  ;                              VRTX workspace calculation:                   *
42  ;  Name |  Description                                    | Value           *
43  ;  -----+-------------------------------------------------+---------------- *
44  ;  **   |  VRTX system variables                          | 256 (fixed)     *
45  ;  t    |  Maximum number of tasks in system              | 13              *
46  ;  us   |  User stack size                                | 8               *
47  ;  p    |  Number of memory partitions in system          |                 *
48  ;  b    |  Number of memory blocks in system              |                 *
49  ;  q    |  Number of queues in system                     |                 *
50  ;  qe   |  Number of queue elements in system             |                 *
51  ;  s    |  Separate interrupt stack flag                  | 0 (unused)      *
52  ;  is   |  Interrupt stack flag                           | 0 (unused)      *
53  ;                                                                            *
54  ;               256 + 48t + 10p + 6b + 12q + 4qe + s                         *
```

```
0000                                    109   TAB       LABEL   BYTE
0000  00 00 00 00                       110             DD      0000H              ;reserved by 8086-- not used here
0004  0000                             111             DW      TRC_SS_ADDR_IP     ;inst. ptr. of entry point for TRACER
                                        112                                        ;single step interrupts
0006  F180                             113             DW      TRACER_CODE_SEG    ;code segment of above
0008  00 00 00 00                       114             DD      0000H              ;reserved by 8086-- not used here
000C  0000                             115             DW      TRC_BP_ADDR_IP     ;inst. ptr. of entry point for TRACER
                                        116                                        ;breakpoint interrupts
000E  F180                             117             DW      TRACER_CODE_SEG    ;code segment of above
0010  0000                             118             DW      0000H              ;reserved by 8086-- not used here
0012     02 [                          119             DD      2 DUP(?)           ;unused
              ????????                  119
                      ]                 119
001A  008F ---- R                      120             DD      CHAR_RECD          ;pointer to data received handler
001E  00D8 ---- R                      121             DD      CHAR_TRANS         ;pointer to data received handler
0022  00 00 00 00                       122             DD      0000H              ;timer #2 interrupt-- not used
0026  0054 ---- R                      123             DD      CLOCK_INT          ;pointer to clock interrupt handler
002A  00 00 00 00                       124             DD      0000H              ;timer #0 interrupt-- not used
002E     13 [                          125             DD      19 DUP(?)          ;unused
              ????????                  125
                      ]                 125
007A  0000                             126             DW      VRTX_IP            ;VRTX entry point
007C  F000                             127             DW      VRTX_CS            ;VRTX entry point
007E     5E [                          128             DD      94 DUP(?)          ;unused
              ????????                  128
                      ]                 128
01F6  00 00 00 00                       129             DD      CFTBL              ;pointer to configuration table
01FA  0000                             130             DW      TRC_ENTRY_IP       ;TRACER entry point IP
01FC  F180                             131             DW      TRACER_CODE_SEG    ;TRACER entry point CS
01FE     7F [                          132             DD      127 DUP(?)         ;unused
              ????????                  132
                      ]                 132
03FA                                   133   VEC_TBL ENDS
                                       134
                                       135
```

```
= 0021                  188   XON             EQU     021H    ;resume output
= 0080                  189   RXRDY           EQU     080H    ;used to check USR to see if data received
= 0080                  190   RXRDY_MASK      EQU     080H    ;masks out lower 7 bits of USR
                        191
                        192   ;*****************************************************************************
                        193   ;                                                                            *
                        194   ;       This next block of EQU's defines the various return codes for calls  *
                        195   ;  used in this board- support- package.                                      *
                        196   ;                                                                            *
                        197   ;*****************************************************************************
                        198
= 0000                  199   RET_OK          EQU     000H            ;successful return--
                        200                                           ;for all commands
= 0001                  201   ER_TID          EQU     001H            ;task ID error
                        202                                           ;for SC_TCREATE
= 0002                  203   ER_TCB          EQU     002H            ;no TCB's available
                        204                                           ;for SC_TCREATE
= 0005                  205   ER_MIU          EQU     005H            ;mailbox in use
                        206                                           ;for SC_POST
= 0006                  207   ER_ZMW          EQU     006H            ;zero message
                        208                                           ;for SC_POST
= 0007                  209   ER_BUF          EQU     007H            ;buffer full
                        210                                           ;for UI_RXCHR
= 000A                  211   ER_TMO          EQU     00AH            ;time- out
                        212                                           ;for SC_PEND
= 000F                  213   ER_INI          EQU     00FH            ;fatal initialization error
                        214                                           ;for VRTX_INIT
= 0010                  215   ER_NCP          EQU     010H            ;no character present
                        216                                           ;for UI_TXRDY
                        217
                        218
                        219
                        220
0001                    221   INIT            PROC    FAR
                        222
                        223                           ;*** stack for VRTX_INIT ***
                        224
0001  C6 06 0000 R 00  225   ENTRY:          MOV     PROTOCOL_STAT,CASE_0    ;initialize XON/XOFF flag
0006  BC 0200 R        226                   MOV     SP,OFFSET TOS           ;set up temporary stack
0009  B8  ---- R       227                   MOV     AX,STACK_SEG            ;of 64-bytes for
000C  8E D0            228                   MOV     SS,AX                   ;VRTX initialization
                        229
                        230                           ;*** call VRTX_INIT ***
                        231
000E  B8 0030          232                   MOV     AX,VRTX_INIT            ;get VRTX_INIT entry point
0011  CD 20            233                   INT     VRTX                    ;call VRTX
0013  3D 0000          234                   CMP     AX,0                    ;any errors?
0016  74 05            235                   JZ      P_INIT
                        236
                        237
                        238
                        239                           ;*****************************************
                        240                           ;***                                   ***
                        241                           ;*** error processing normally done here ***
```

```
;*****************************************************************************
;
;   This block will initialize the 8254 Programmable Interval Timer. This   *
; device is being used for the following purpose:                           *
;     -- To generate interrupts every 10ms for the VRTX system clock         *
;   Programming the 8254 involves two basic steps:                           *
;     1) Write a control word to the 8254 which will inform it as to what    *
;        information will be sent next, i.e. which timer, how the count      *
;        is to be sent, etc.                                                 *
;     2) Write the initial count value to the proper timer register          *
;   The control word bits specify the following:                             *
;                         b7 b6 b5 b4 b3 b2 b1 b0                             *
;                     sc1__|  |  |  |  |  |  |  |                             *
;                     sc0_____|  |  |  |  |  |  |                             *
;                     rw1________|  |  |  |  |  |                             *
;                     rw0___________|  |  |  |  |                             *
;                      m2______________|  |  |  |                             *
;                      m1_________________|  |  |                             *
;                      m0____________________|  |                             *
;                     bcd_______________________|                             *
;   Description:                                                             *
;      sc1,sc0 ...timer select                                               *
;         00 => select timer #0                                              *
;         01 => select timer #1                                              *
;         10 => select timer #2                                              *
;         11 => allows for read-back of current count                        *
;      rw1,rw0 ...how init count is to be written into the timer register    *
;         00 => counter latch command                                        *
;         01 => Read/Write least significant byte only                       *
;         10 => Read/Write most significant byte only                        *
;         11 => Read/Write least significant, then most significant          *
;      m2,m1,m0 ..determines which mode the timer will operate in            *
;         000 => mode 0: interrupt on terminal count                         *
;         001 => mode 1: hardware retriggerable one-shot                     *
;         x10 => mode 2: rate generator                                      *
;         x11 => mode 3: square wave mode                                    *
;         100 => mode 4: software triggered mode                             *
;         101 => mode 5: hardware triggered strobe                           *
;      bcd .......determines if the counter will be BCD or 16-bit binary     *
;         0 => normal 16-bit binary counter                                  *
;         1 => BCD counter, 4 decades                                        *
;    Summary:                                                                *
;      Through the use of the select bits, each timer may be individually    *
;      accessed and set up accordingly. Once the control words have set      *
;      up the timer operation, then the initial count values may be writ-    *
;      ten to the appropriate timer register. On the Seldin-1, the ports     *
;      are as follows:                                                       *
;         Control Register =  $40                                            *
;         Timer #0 Register = $42                                            *
;         Timer #1 Register = $44                                            *
;         Timer #2 Register = $46                                            *
;      Thus, the programming sequence is as follows:                         *
;         1) Set up timer
```

```
0050  B8 0030                  390                   MOV     AX,PIT_1_MSB      ;get msb for timer #1
0053  EF                       391                   OUT     DX,AX             ;initialize msb of co
0054                           392   PIT_INIT        ENDP
                               393
```

0064

```
448
449   CLOCK_INT       ENDP
450
451   ;*****************************************************************************
452   ;                                                                            *
453   ;       Initialization for the 8259 Programmable Interrupt Controller        *
454   ;                                                                            *
455   ;       The 8259 PIC requires the following data to be programmed into it    *
456   ;  before it will operate correctly:                                         *
457   ;          - is there only one PIC, or are there more?                       *
458   ;          - what is the address interval of the IVT?                        *
459   ;          - what is the begining address of the interrupt vector table?     *
460   ;          - is the CPU an 8086/8088 or an 8085?                             *
461   ;          - auto or normal end of interrupt?                                *
462   ;          - fully nested or not fully nested mode?                          *
463   ;       This information is given to the PIC via initialization words:       *
464   ;          - ICW1                                                            *
465   ;          - ICW2                                                            *
466   ;          - ICW3 if in cascaded mode                                        *
467   ;          - ICW4                                                            *
468   ;       I will set up the PIC as follows:                                    *
469   ;          - single mode                                                     *
470   ;          - address interval of 4 bytes                                     *
471   ;          - edge triggered mode                                             *
472   ;          - 8086 mode                                                       *
473   ;          - normal end-of-interrupt                                         *
474   ;          - special fully-nested mode                                       *
475   ;       Note: I do not need ICW3                                             *
476   ;                                                                            *
477   ;       To begin initialization of the PIC, the first initialization word    *
478   ;       must be written to the lower address. It is recognized as the        *
479   ;       start of the initialization process when A0 = 0 and D4 = 1. Once     *
480   ;       the first ICW has been recognized, the PIC looks for the rest of     *
481   ;       the sequence in the following order:                                 *
482   ;          1) ICW2                                                           *
483   ;          2) ICW3, if in cascaded mode (indicated in ICW1)                  *
484   ;          3) ICW4, if specified in ICW1                                     *
485   ;       All initialization control words, other than the first, will go to   *
486   ;       the higher address. Once the PIC is initialized, operation control   *
487   ;       words may be sent at any time, and as many times as desired. The     *
488   ;       OCWs are distinguished from one another as follows:                  *
489   ;             - OCW1 is any word sent to the higher address after initial-   *
490   ;               ization.                                                     *
491   ;             - OCW2 is any word sent after initialization with A0 = 0,      *
492   ;               D4 = 0, and D3 = 0.                                          *
493   ;             - OCW3 is any word sent after initialization with A0 = 0,      *
494   ;               D4 = 0, and D3 = 1.                                          *
495   ;                                                                            *
496   ;*****************************************************************************
497
498   ;*****************************************************************************
499   ;                                                                            *
500   ;                   Calculations for ICWs and OCWs                           *
501   ;                   ------------------------------                           *
```

```
548
549  ;*****************************************************************************
550  ;                                                                             *
551  ;                        Interrupt Vector Table structure                     *
552  ;                        --------------------------------                     *
553  ;    Int. # (type) |          Source                |   IVT location (IP:CS)  *
554  ; -----------------+--------------------------------+-------------------------*
555  ;          0       |   Divide by zero error         |          00             *
556  ;          1       |   Single step                  |          04             *
557  ;          2       |   Non- maskable interrupt      |          08             *
558  ;          3       |   One- byte interrupt          |          0C             *
559  ;          4       |   Interrupt on overflow        |          10             *
560  ;          .       |                                |          .              *
561  ;          .       |   Unused                       |          .              *
562  ;          .       |                                |          .              *
563  ;          7       |   UART-- data received         |          20             *
564  ;          8       |   UART-- transmit completed    |          24             *
565  ;          9       |   PIC-- timer #2 interrupt     |          28             *
566  ;          10      |   PIC-- timer #1 interrupt     |          2C             *
567  ;          11      |   PIC-- timer #0 interrupt     |          30             *
568  ;          .       |                                |          .              *
569  ;          .       |   Unused                       |          .              *
570  ;          .       |                                |          .              *
571  ;          32      |   VRTX entry pointer           |          80             *
572  ;          .       |                                |          .              *
573  ;          .       |   Unused                       |          .              *
574  ;          .       |                                |          .              *
575  ;          128     |   Configuration table ptr.     |          200            *
576  ;          129     |   Pointer to TRACER entry      |          204            *
577  ;          .       |                                |          .              *
578  ;          .       |   Unused                       |          .              *
579  ;          .       |                                |          .              *
580  ;          255     |   Unused                       |          3FF            *
581  ;                                                                             *
582  ;    NOTE:                                                                    *
583  ;      - The first five interrupts (0 - 4) are reserved by the 8086           *
584  ;                                                                             *
585  ;*****************************************************************************
586  ;                                                                             *
587  ;                How to calculate the Interrupt Vectors                       *
588  ;                --------------------------------------                       *
589  ;   First of all, the base address of the IVT is programmed into the PIC      *
590  ; via ICW2. All of the PIC's address calculations are made relative to this. *
591  ; The PIC on the Seldin-1 incorporates 7 of 8 interrupts (if more necessary   *
592  ; more PICs can be cascaded, yielding up to 64 hardware interrupts).         *
593  ;    Lowest priority    I0 = UART/ data received                              *
594  ;                       I1 = UART/ data transmission complete                 *
595  ;                       I2 = third timer interrupt from the PIC               *
596  ;                       I3 = second timer interrupt                           *
597  ;                       I4 = first timer interrupt                            *
598  ;                       I5 = bad address interrupt                            *
599  ;                       I6 = end conv (from ALP)                              *
600  ;    Highest priority   I7 = not used                                         *
601  ; When one or more of the interrupt requests are generated, the PIC does     *
```

```
634
635 ;*****************************************************************************
636 ;                                                                            *
637 ;                    Initialization for the 8252 UART                        *
638 ;                    --------------------------------                        *
639 ;         Only three control registers need to be initialized on the 8252:   *
640 ;              1) UART Control Register (UCR)                                *
641 ;              2) Baud Rate Selector Register (BRSR)                         *
642 ;              3) Modem Control Register (MCR)                               *
643 ;         The UCR sets up the following:                                     *
644 ;              1) word length                                                *
645 ;              2) parity                                                     *
646 ;              3) number of stop bits                                        *
647 ;         The BRSR computes the baud rate given the following:               *
648 ;              1) prescaler select-- divides the clock frequency             *
649 ;              2) divisor select-- divides the result further into a baud rate*
650 ;              3) CO select-- determines if the CO pin of the UART will out- *
651 ;                 put the external clock frequency or the baud rate generator.*
652 ;         The MCR is used for the following:                                 *
653 ;              1) control of the output of the following pins on the 8252:   *
654 ;                   a. RTS ..... request to send                             *
655 ;                   b. DTR ..... data terminal ready                         *
656 ;                   c. INTEN .... interrupt enable                           *
657 ;                   d. REN ..... receiver enable                             *
658 ;                   e. MIEN ..... modem interrupt enable                     *
659 ;              2) selects mode:                                              *
660 ;                   a. normal                                                *
661 ;                   b. transmit break                                        *
662 ;                   c. echo mode                                             *
663 ;                   d. loop test mode                                        *
664 ;         The Seldin-1 has the following addresses for the ports:            *
665 ;              1) $30 => data I/O                                            *
666 ;              2) $32 => UART control and status registers                   *
667 ;              3) $34 => modem control register                              *
668 ;              4) $36 => bit rate select register and modem status register  *
669 ;         Thus, each address corresponds to two possible operations. Possible*
670 ;         instructions:                                                      *
671 ;              - OUT nn --> $30 outputs a character to the serial port       *
672 ;              - IN  AX <-- $30 will input a character from the serial port  *
673 ;              - OUT nn --> $32 initializes the UCR                          *
674 ;              - IN  AX <-- $32 reads the UART Status Register               *
675 ;              - OUT nn --> $34 initializes the MCR                          *
676 ;              - IN  AX <-- $34 reads the MCR                                *
677 ;              - OUT nn --> $36 selects the baud rate                        *
678 ;              - IN  AX <-- $36 reads the Modem Status Register              *
679 ;         I will use the following to initialize the UCR, MCR, and BRSR:     *
680 ;              UCR_CTRL:                                                     *
681 ;                   - D7 - D6 are reserved for future use-- set to 0         *
682 ;                   - D5 - D4 = 10 for 7-bit characters                      *
683 ;                   - D3 - D1 = 000 for even parity                          *
684 ;                   - D0 = 0 for one stop bit                                *
685 ;              MCR_CTRL:                                                     *
686 ;                   - D7 = 0 for normal operation                            *
687 ;                   - D6 = 0 to disable modem interrupt                      *
```

```
;     1 = output stopped, but no transmit interrupt has occurred
;     2 = output stopped, and a transmit interrupt has occurred
; I use the standard <ctrl> s as XOFF and <ctrl> q as XON
; The pseudo code for the routines can be found in the system release notes,
; but in case they have disappeared, I will include it here. The flowcharts
; for these routines should be around somewhere in my external documentation
;
;    RECEIVE:
;         get character from UART
;         if flag is 0 then
;            if character is <ctrl> s then
;               set flag to 1
;            else
;               post character with UI_RXCHR
;            endif
;         else
;            if character is <ctrl> q then
;               if flag is 2 then
;                  do transmit_char
;               endif
;               set flag to 0
;            else
;               post character with UI_RXCHR
;            endif
;         endif
;
;    TRANSMIT:
;         if flag is 0 then
;            do transmit_char
;         else
;            set flag = 2
;         endif
;
;    TRANSMIT_CHAR:
;         get character from VRTX using UI_TXRDY
;         output character to UAR
;
;*****************************************************************************


;*****************************************************************************
;
;                  Character received interrupt handler
;                  ------------------------------------
;     This procedure will be accessed whenever the DR pin on the 8252 UART
;  becomes active. The handler uses the following calls:
;        1) UI_ENTER/EXIT as usual
;        2) UI_RXCHR:
;             ENTRY:
;                AX = UI_RXCHR (0013H)
;                CH = character
;             EXIT:
;                AX = return code
;
```

```
                                ;         AX = return code                                      *
                                ;         CH = character                                        *
                                ;  Therefore, the handler simply requests the next character to be printed, *
                                ;  and then the TXRDY driver will output the character to the appropriate  *
                                ;  device.                                                       *
                                ;*****************************************************************

00D8                            CHAR_TRANS      PROC    FAR

                                                ;*** enter system mode ***

00D8  50                                        PUSH    AX              ;save for UI_EXIT to restore

                                                ;*** request next character to be printed ***

00D9  80 3E 0000 R 00                           CMP     PROTOCOL_STAT,CASE_0
00DE  75 0D                                     JNE     SETFLAGG
00E0  B8 0014                                   MOV     AX,UI_TXRDY
00E3  CD 20                                     INT     VRTX
00E5  9A 0106 ---- R                            CALL    FAR PTR TXRDY
00EA  EB 15 90                                  JMP     EXT
00ED  C6 06 0000 R 02           SETFLAGG:       MOV     PROTOCOL_STAT,CASE_2

00F2  B8 0014                                   MOV     AX,UI_TXRDY
00F5  CD 20                                     INT     VRTX            ;call VRTX
00F7  3D 0000                                   CMP     AX,0            ;return code OK?
00FA  75 05                                     JNZ     EXT             ;if not, exit

                                                ;*** call driver to transmit character ***

00FC  9A 0106 ---- R                            CALL    FAR PTR TXRDY   ;go to transmit it

                                                ;*** exit system mode ***

0101  B8 0011                   EXT:            MOV     AX,UI_EXIT      ;exit
0104  CD 20                                     INT     VRTX            ;call VRTX-- VRTX restores AX
0106                            CHAR_TRANS      ENDP
                                                PUBLIC  TXRDY
0106                            TXRDY           PROC    FAR
0106  8A C5                                     MOV     AL,CH           ;get character
0108  E7 30                                     OUT     UART_DATA,AX    ;send the char. to the output register
010A  CB                                        RET                     ;return to whoever called
010B                            TXRDY           ENDP


                                ;*****************************************************************
                                ;                                                               *
                                ;       This is the polled- downloading routine. This operates with inter-  *
                                ; rupts turned off, so as to avoid possible data transmission errors and to *
                                ; improve the speed of transmission. When the transfer is complete, inter-  *
                                ; rupts are turned back on.                                     *
                                ;                                                               *
                                ;*****************************************************************
```

```
                                958
                                959                 ;*** output a ">" prompt to CRT ***
                                960
0019  B5 3E                     961                 MOV     CH,PROMPT       ;get a ">"
001B  B8 000E                   962                 MOV     AX,SC_PUTC      ;put the char. in mailbox
001E  CD 20                     963                 INT     VRTX            ;call VRTX
                                964
                                965                 ;*** clear message registers ***
                                966
0020  2B C9                     967                 SUB     CX,CX           ;clear CX (message word 1)
0022  2B D2                     968                 SUB     DX,DX           ;clear DX (message word 2)
                                969
                                970                 ;*** get a character from the CRT ***
                                971
0024  B8 000D                   972   GETC:         MOV     AX,SC_GETC      ;get a character
0027  CD 20                     973                 INT     VRTX            ;call VRTX
                                974
                                975                 ;*** post character in mailbox ***
                                976
0029  B8 0008                   977                 MOV     AX,SC_POST      ;post it
002C  CD 20                     978                 INT     VRTX            ;call VRTX
                                979
                                980                 ;*** go to get next character ***
                                981
002E  EB F4                     982                 JMP     GETC
0030                            983   MAIN          ENDP
                                984
                                985   ;*****************************************************************************
                                986   ;                                                                           *
                                987   ;    This task picks up the character to be printed from the mailbox and    *
                                988   ; then outputs it to the CRT. It makes the following call: SC_PEND:        *
                                989   ;               ENTRY:                                                      *
                                990   ;                  AX = SC_PEND (0009H)                                     *
                                991   ;                  CX = time-out value (low-order word)                     *
                                992   ;                  DX = time-out value (high-order word)                    *
                                993   ;                  ES:BX = address of mailbox                               *
                                994   ;               EXIT:                                                       *
                                995   ;                  AX = return code                                         *
                                996   ;                  CX = message (low-order word)                            *
                                997   ;                  DX = message (high-order word)                           *
                                998   ;                                                                           *
                                999   *****************************************************************************
                               1000
0030                           1001   TASK          PROC    NEAR
                               1002                         ;*** set up mailbox context ***
                               1003
0030  BB 0000 R                1004                 MOV     BX,OFFSET MBOX          ;get mailbox offset
0033  B8  ---- R               1005                 MOV     AX,SEG MBOX             ;get mailbox sement
0036  8E C0                    1006                 MOV     ES,AX                   ;put segment in ES
                               1007
                               1008                 ;*** pend to get character from mailbox ***
                               1009
0038  2B C9                    1010   PENDC:        SUB     CX,CX                   ;no timeout
003A  2B D2                    1011                 SUB     DX,DX                   ;no timeout
```

Segments and groups:

Name	Size	align	combine	class
BSP_CODE	0117	BYTE	PUBLIC	'BSP_CO'
CFTBL	0030	PARA	NONE	'DATA'
CODE	0048	PARA	NONE	'CODE'
MAIL	0004	PARA	NONE	'DATA'
STACK_SEG	0200	PARA	NONE	
TCONFIG	0038	PARA	NONE	'DATA'
VEC_TBL	03FA	PARA	NONE	'VECTOR'

Symbols:

Name	Type	Value	Attr		
BRSR_ADDR	Number	0036			
BRSR_CTRL	Number	0086			
CASE_0	Number	0000			
CASE_1	Number	0001			
CASE_2	Number	0002			
CHANNELS	Number	0001			
CHAR_RECD	F PROC	008F	BSP_CODE		Length =0049
CHAR_TRANS	F PROC	00D8	BSP_CODE		Length =002E
CHECK_XON	L NEAR	00AC	BSP_CODE		
CLOCK_INT	F PROC	0054	BSP_CODE		Length =0010
DNLD	F PROC	010B	BSP_CODE		Length =000C
ENTRY	L NEAR	0001	BSP_CODE		
ER_BUF	Number	0007			
ER_INI	Number	000F			
ER_MIU	Number	0005			
ER_NCP	Number	0010			
ER_TCB	Number	0002			
ER_TID	Number	0001			
ER_TMO	Number	000A			
ER_ZMW	Number	0006			
EXT	L NEAR	0101	BSP_CODE		
GETC	L NEAR	0024	CODE		
ICW1	Number	0017			
ICW2	Number	0000			
ICW4	Number	001F			
INIT	F PROC	0001	BSP_CODE		Length =0041
IO_TOGGLE	Number	001B			
ISSUE_TO_VRTX	L NEAR	00C9	BSP_CODE		
MAIN	N PROC	0000	CODE	Global	Length =0030
MBOX	L DWORD	0000	MAIL		
MCR_ADDR	Number	0034			
MCR_CTRL	Number	0006			
OCW1	Number	0000			
PENDC	L NEAR	0038	CODE		
PIC_HI_ADDR	Number	0022			
PIC_INIT	F PROC	0064	BSP_CODE		Length =0016
PIC_LO_ADDR	Number	0020			
PIT_0_ADDR	Number	0042			

XON. . . . . Number 0021

Warning Errors	Severe Errors
0	0

```
FS MicroSET-86  LINKER  V1.93.00    14-MAY-1987  11:25:54

OUTPUT FILE: SUPPORT_V2.LNK


INVOKED BY:
        FSLINK SUPPORT_V2


Processing input file SUPPORT_V2
Processing input file SUPPORT_V2

  ......NAME.....   KIND OF OBJECT    FIRST DEFINITION/USE

  A                 MODULE
  TAB               PUBLIC            A
  SUPPORT_V2        FILE
  BSP_CODE          CLASS
                    SEGMENT           A
  TABL              PUBLIC            A
  DATA              CLASS
  CODE              CLASS
                    SEGMENT           A
  MAIN              PUBLIC            A
  STACK_SEG         SEGMENT           A
  VEC_TBL           SEGMENT           A
  MAIL              SEGMENT           A
  VECTOR            CLASS
  CFTBL             SEGMENT           A
  TXRDY             PUBLIC            A
  TCONFIG           SEGMENT           A
  TBL               PUBLIC            A


END OF MAP
```

THE UNIVERSITY OF MICHIGAN

MEMORANDUM

September 18, 1990

MEMO TO: Advanced Langmuir Probe Project File (#021546)

FROM: B. P. Block

SUBJECT: Report on the Implementation of a High Frequency Switching Power Converter for the Advanced Langmuir Probe

Introduction

The static Langmuir probe technique, as it has been adapted for the measurement of the characteristics of ionospheric plasmas, employs a cylindrical collector that is immersed in the plasma and electrically stepped through a range of applied voltage relative to the spacecraft (and plasma). A resultant current is induced in the collector and subsequently measured by a sensitive electrometer, allowing plasma density and temperature to be deduced from the familiar volt-ampere curve. The electrometer and applied voltage generator circuitry may be configured in a variety of topologies depending upon the nature of the measurement and the requirements of the particular space vehicle. Most Langmuir probe instruments built at this laboratory have used a configuration in which the electrometer circuitry is biased (or floating) atop the the applied voltage (V_A). This approach was no doubt originally adopted because it leads to a simplification in the V_A generator circuitry, which can operate against spacecraft ground, and because it allows the cylindrical probe to be connected directly to the electrometer input. These instruments however have shared a problem associated with this topology: the adverse effects of switching power converter noise currents introduced into the V_A-probe-plasma loop by the floating electrometer power supply. The focus of this part of the Advanced Langmuir Probe study conducted under NASA grant NAG5-419 has been to design, fabricate and test the electronics and software for reducing the effects of these noise currents to a level at which

they have no sensible effect on the measurement. This report describes the design of this system and discusses some of the results obtained.

The Effect of Induced Noise in the V_A-Probe-Plasma Loop

A simplified view of the electrometer and voltage generator in their traditional configuration is shown in Figure 1. The floating electrometer requires supply voltages that are separate from those of the V_A generator and control logic; these floating potentials are generally derived from separate secondaries on the main DC-DC power converter. These supplies are not shown for the sake of simplicity, but the noise currents that they induce through primary-secondary capacitive coupling are shown as current source I_{PSN}.

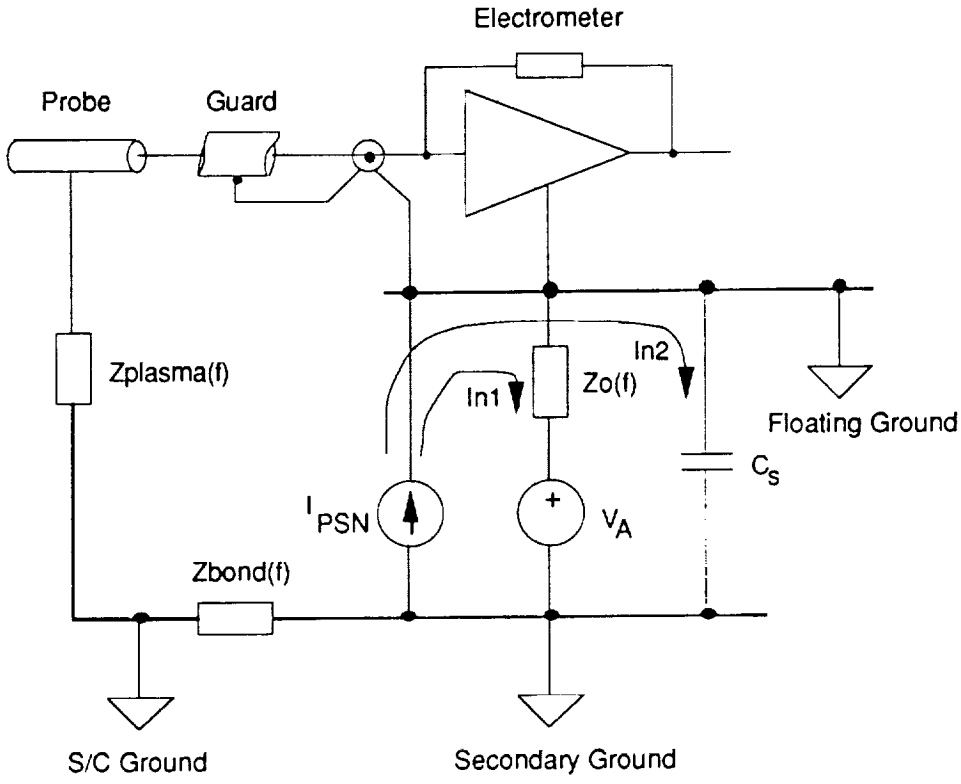


Figure 1. Electrometer and V_A Generator Concept

Even though considerable care is exercised in reducing stray capacitive coupling from the primary winding to the floating windings through use of

Faraday shielding and other techniques, noise currents at the converter frequency (generally in the range of several tens of kiloHertz) and its harmonics are inevitably coupled, directly or indirectly, into the floating electrometer circuitry. These currents are illustrated as I_{n1} and I_{n2}. If the output impedance of the V_A generator could be made zero, or at least very small with respect to the plasma impedance, the switching noise currents I_{n1} would be shunted through the generator without perturbing the applied potential. However, practical fed-back voltage generators, which are capable of developing precise applied potentials, will always have finite output impedances as indicated by $Z_0(f)$ in Figure 1. Noise currents (at the frequency of the switching power supply) in the microampere range are quite capable of inducing voltages in the range of many millivolts, causing substantial errors in the observed plasma current. A second current path exists through the distributed capacitance between the electrometer shield, which can be extensive, and the secondary ground. The impedance of this path, purely capacitive, is generally large with respect to that of the V_A generator in the new configuration, and the effects of this current can be neglected in the following discussion.

The effect of I_{n1} as shown is to introduce a momentary voltage offset, indistinguishable from a step change in V_A, into the V_A-probe-plasma loop. In earlier versions of the Langmuir probe instrument, the response time of the electrometer amplifier to an impulse of applied voltage was considerably slower than the duration of the current transient produced by the above-discussed mechanism. Consequently, no adverse effect on the operation of the electrometer was observed. But the response time of the high-speed electrometer implemented under this Grant (and discussed in a previous report)[1] is significantly faster than earlier versions and is well within the range of transient pulse widths. These effects are most egregious in the electron retardation region, precisely that part of the volt-ampere curve where accurate temperature measurements must be made. That the effect is most prominent in this region is obvious from the fact that the conductance of the plasma ($\partial I_{plasma}/\partial V_A$) is largest in this area.

---

[1] B. P. Block, "Report on the Implementation and Performance of an Improved Langmuir Probe Amplifier", internal memorandum 86-002, Space Physics Research Laboratory, University of Michigan, Ann Arbor, February 13, 1986.

Indeed, a recovery period of about 800 microseconds is required after each transient, which can be seen from measured data.[2] This recovery time coupled with recovery due to overload and saturation of the electrometer amplifier encountered during step changes in V_A lead to slow and imprecise measurements of the plasma characteristics.

A High-Frequency Floating Power Converter

One solution to this dilemma, and the approach taken in this study, implements a high-frequency power converter whose fundamental frequency is several times that of the bandwidth of the current amplifier. A schematic diagram of the converter is shown in Figure 2. This converter could be implemented in a number of technologies, such as simple pulse width modulation or resonant mode conversion; however, a current-mode design was chosen for the sake of lightness and simplicity. The converter operates at approximately 450 kHz, which is a factor of two higher than the measured 3-dB bandwidth of the fastest (and least sensitive) electrometer current range. The converter derives primary power from +28 volts, but could operate from supplies as low as 10 volts. The controller microchip, U1, is designed for high-frequency current-mode operation and drives Q1, a power MOSFET, directly. The transformer, T1, is wound on an RM6-style ferrite pot core, Siemens B65807-C-R47. High-frequency signal diodes are used for secondary rectification. Voltage regulation is obtained by a tightly-coupled secondary whose voltage is rectified and fed-back to the error amplifier in U1. The design is straightforward, presenting little difficulty in the way of test and operation. No attempt was made to electrostatically isolate the secondary from the primary windings.

Observed Results

A conceptual view of the floating electrometer and V_A generator (to be discussed in a separate memorandum) are shown in Figure 3.
Earlier tests of this system were conducted with the floating electrometer operated from a commercial DC-DC converter (at 60 kHz). The results were

---

[2] Ibid., Figure 6a, p. 14.

dramatic. A careful examination of the electrometer preamplifier output (at maximum sensitivity), operated in the electron retardation region of a dummy plasma and observed on a differential oscilloscope to remove V_A, revealed no trace of the floating converter fundamental or its harmonics. It was decided to measure the output of the preamplifier to a precision greater than that afforded by the internal A/D converter, which possesses a resolution of +/- 1 part in 4096. For this purpose, a Hewlett-Packard 3458A digital multimeter with a resolution of 6.5 decimal digits was chosen. A variety of high value resistors in the several hundred megohm range were chosen to provide large plasma impedances. These resistors were placed in a specially designed, shielded box to prevent extraneous noise pickup. The imposed voltage was delivered by the V_A generator.

Finally, software was written for the Advanced Langmuir Probe Dedicated Processor and bench check computer[3]. This software allows the user to sweep the V_A generator over its entire range while selecting the size of each voltage step. The software queries the digital multimeter for the value of the electrometer output voltage and records the two values, voltage step and output current. This has been done over the four preamplifier ranges and with a variety of step sizes. The data were plotted in graphical form and are presented in the Appendix.

Before characterizing the response of the electrometer preamplifier, a series of linearity checks were run on the V_A generator. These data are summarized in Figures A1 and A2. No departure from 16-bit differential linearity was observed.

The conditions under which switching converter noise might identify itself are evidenced in Figure A3 and A4. Here the simulated plasma impedance is large (1000 Megohms). Earlier curves taken under these conditions with the commercial low-frequency power converter were showed large departures from the linear on both the full and reduced ranges. As one can see, no noise is discernable in any part of the voltage

[3] B. P. Block, "Report on the Implementation of a Dedicated Processor for the Advanced Langmuir Probe", internal memorandum 87-008, Space Physics Research Laboratory, University of Michigan, Ann Arbor, June 4, 1987.

sweep. Likewise, the remaining ranges at lower sensitivities yield no noticeable scatter.

Conclusion

Thus it is demonstrated that the effect of switching noise in the floating power converter can be reduced to a level in which it plays no part in the measurement. In fact, we have returned to the case of the earlier Langmuir probe instruments, in which the electrometer response time was slower than the perturbing impulse. A note of caution should be sounded, because in all cases the plasma itself is subjected to an impure applied voltage, even if the instrument itself is unable to respond to the effects of such a perturbation. The plasma, possessing a non-linear volt-ampere characteristic, is of course not immune to such high frequency waveforms, and as such introduces some uncertainty, however small, into the measurement.

Additional Figures

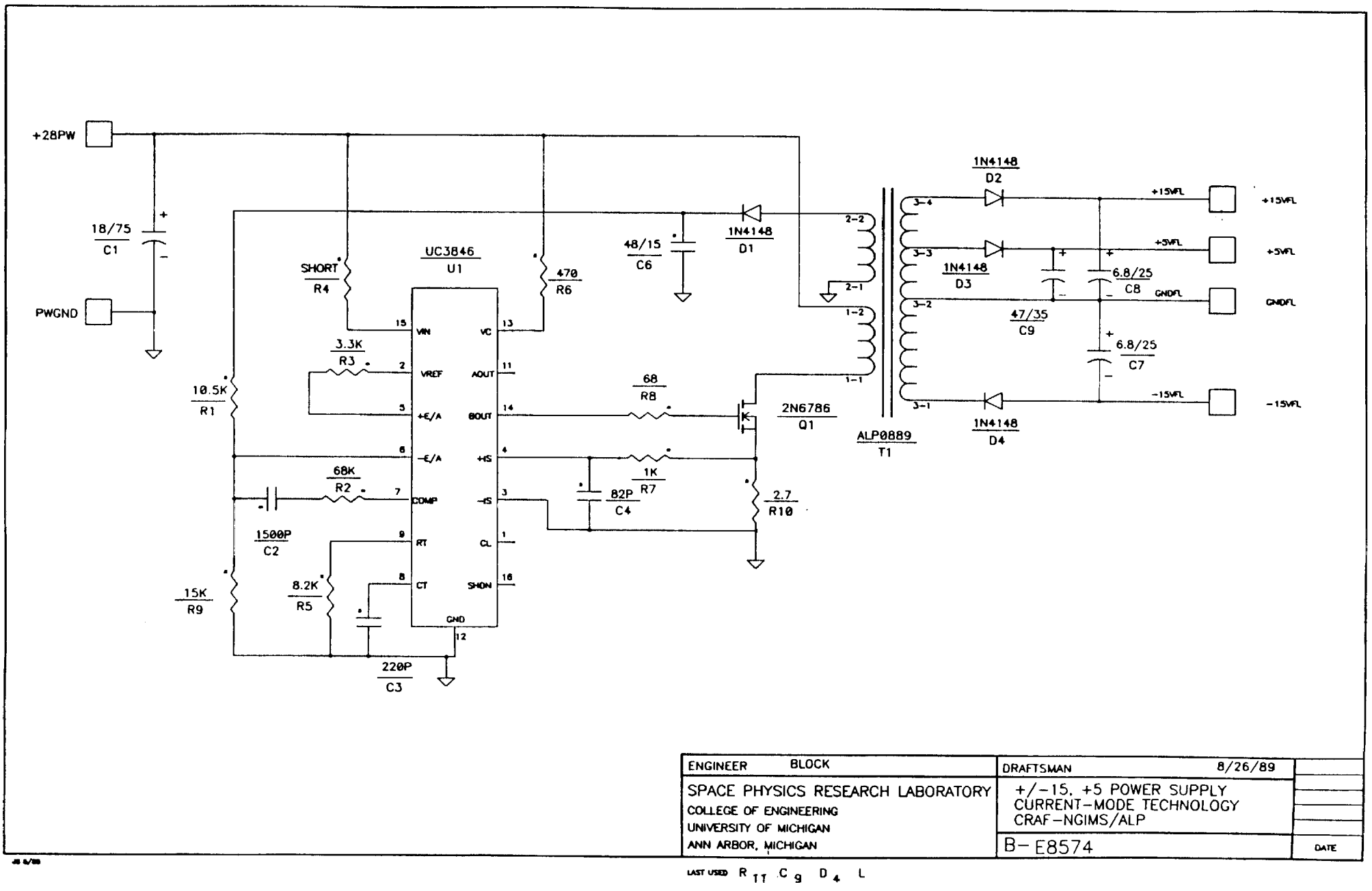


Figure 2. Floating Electrometer Power Supply

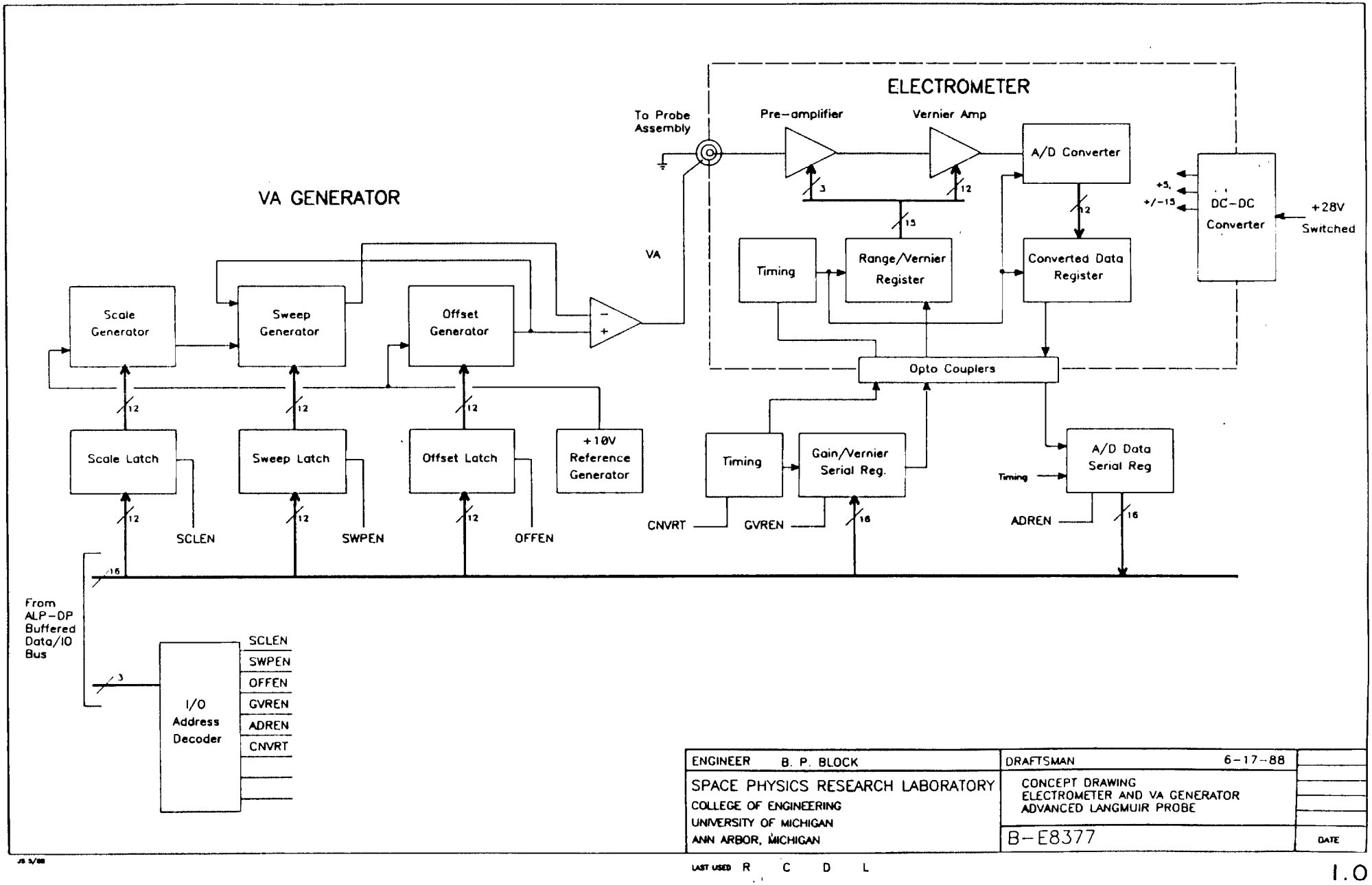


Figure 3. Concept Drawing, Electrometer and VA Generator

Appendix

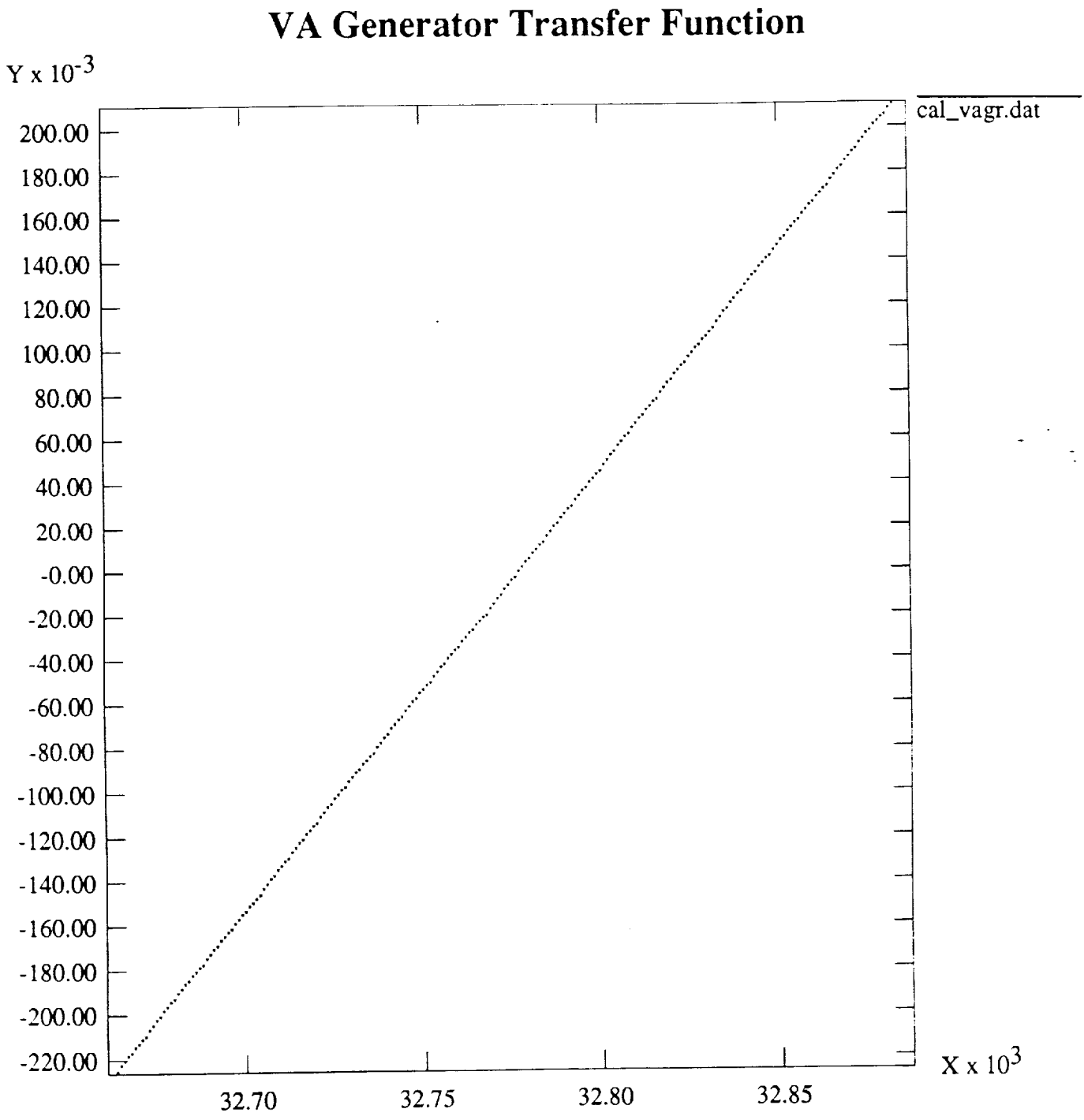


Figure A1. V_A Transfer Function, Full Range

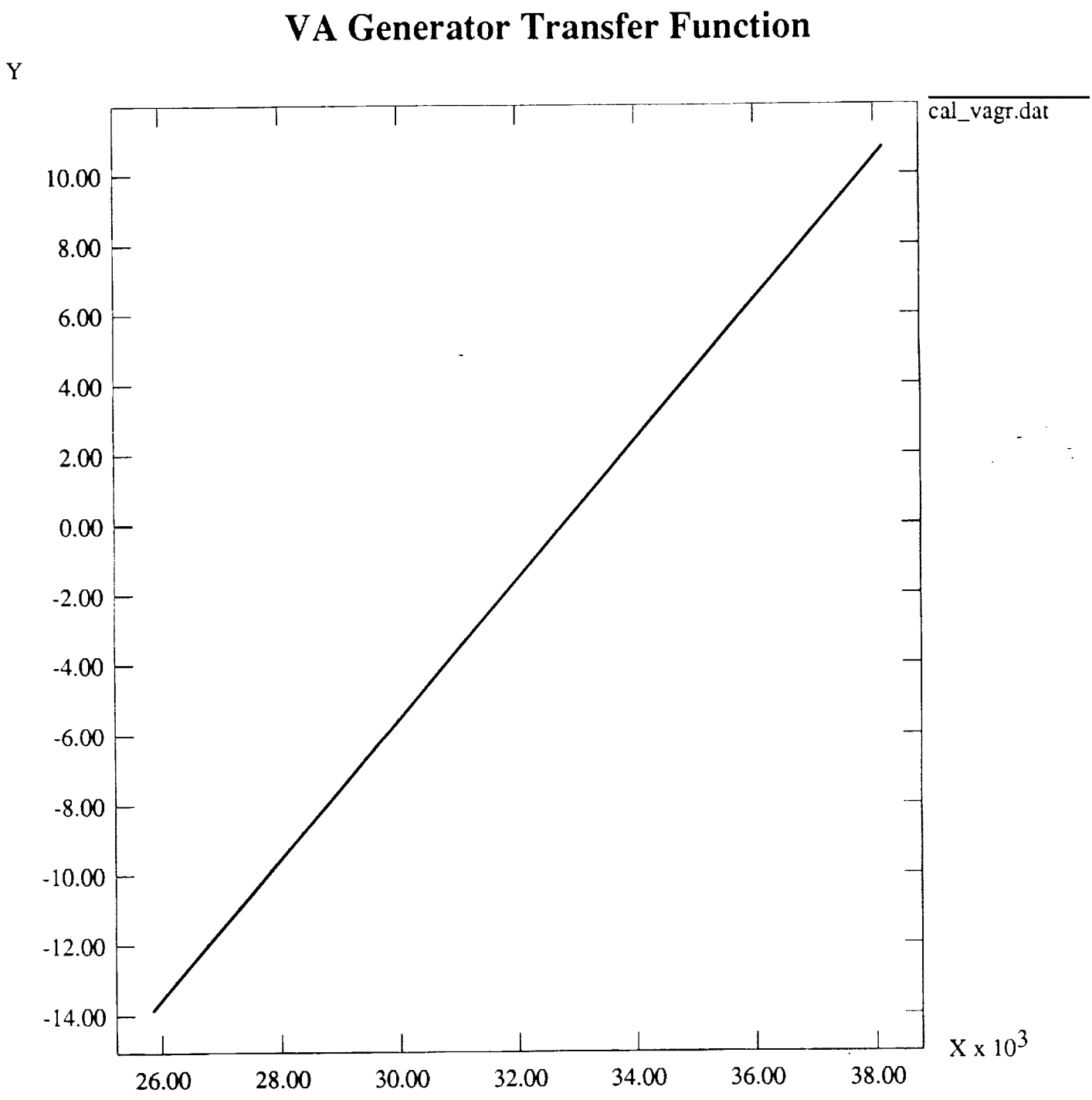


Figure A2. V_A Transfer Function, Reduced Range

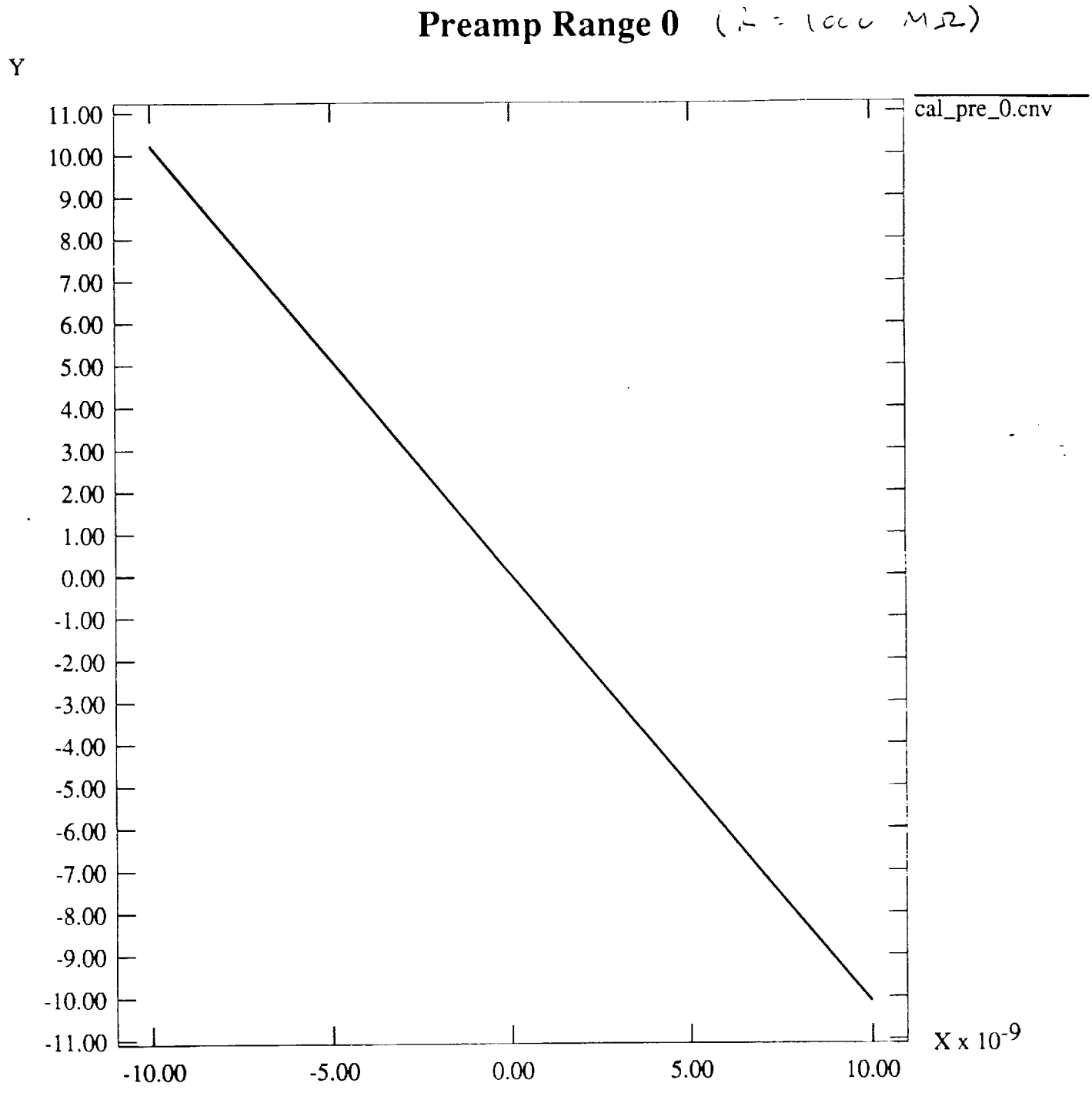


Figure A3. Preamp Range 0, Full Range (R=1000M)

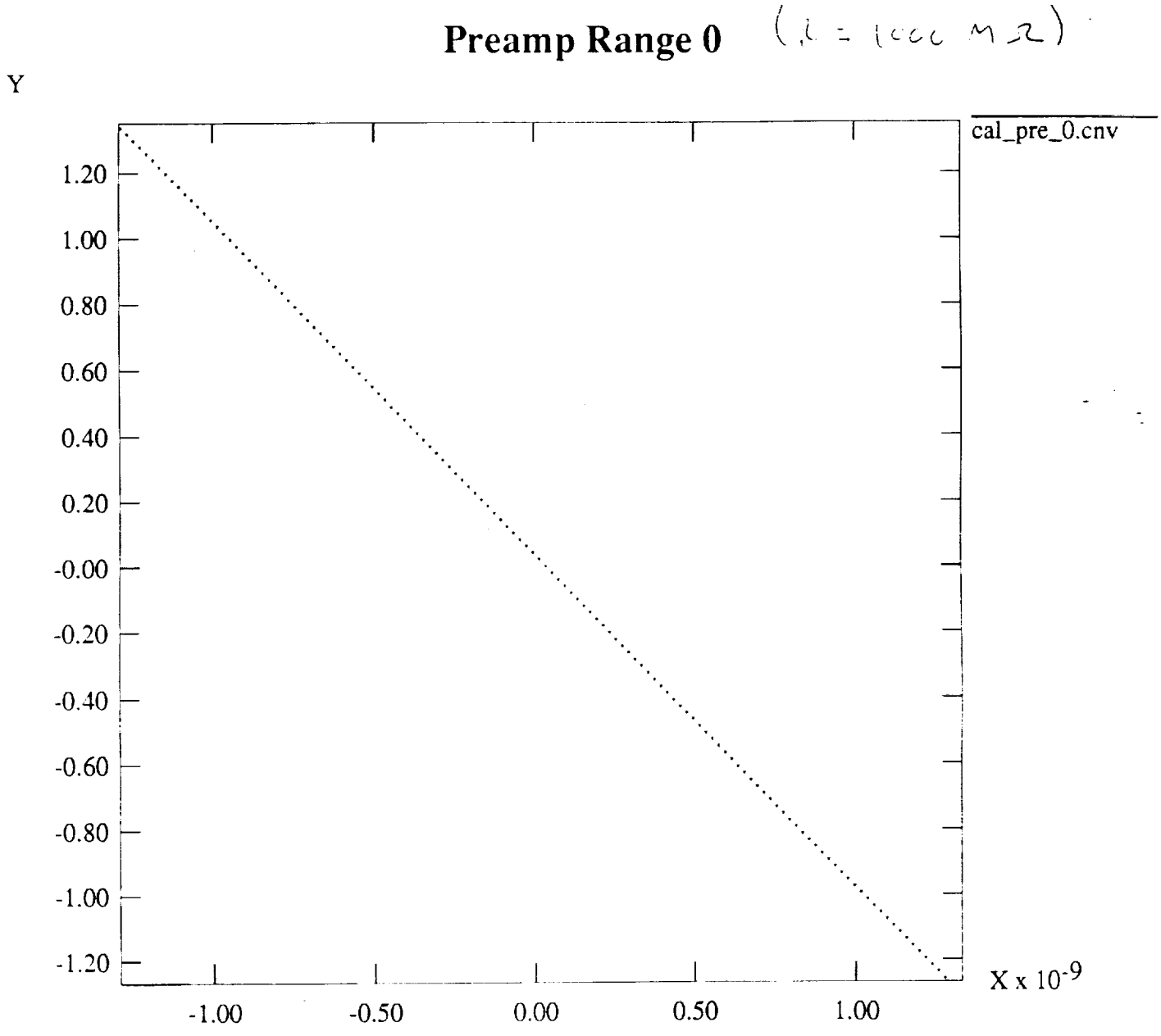


Figure A4. Preamp Range 0, Reduced Range (R=1000M)

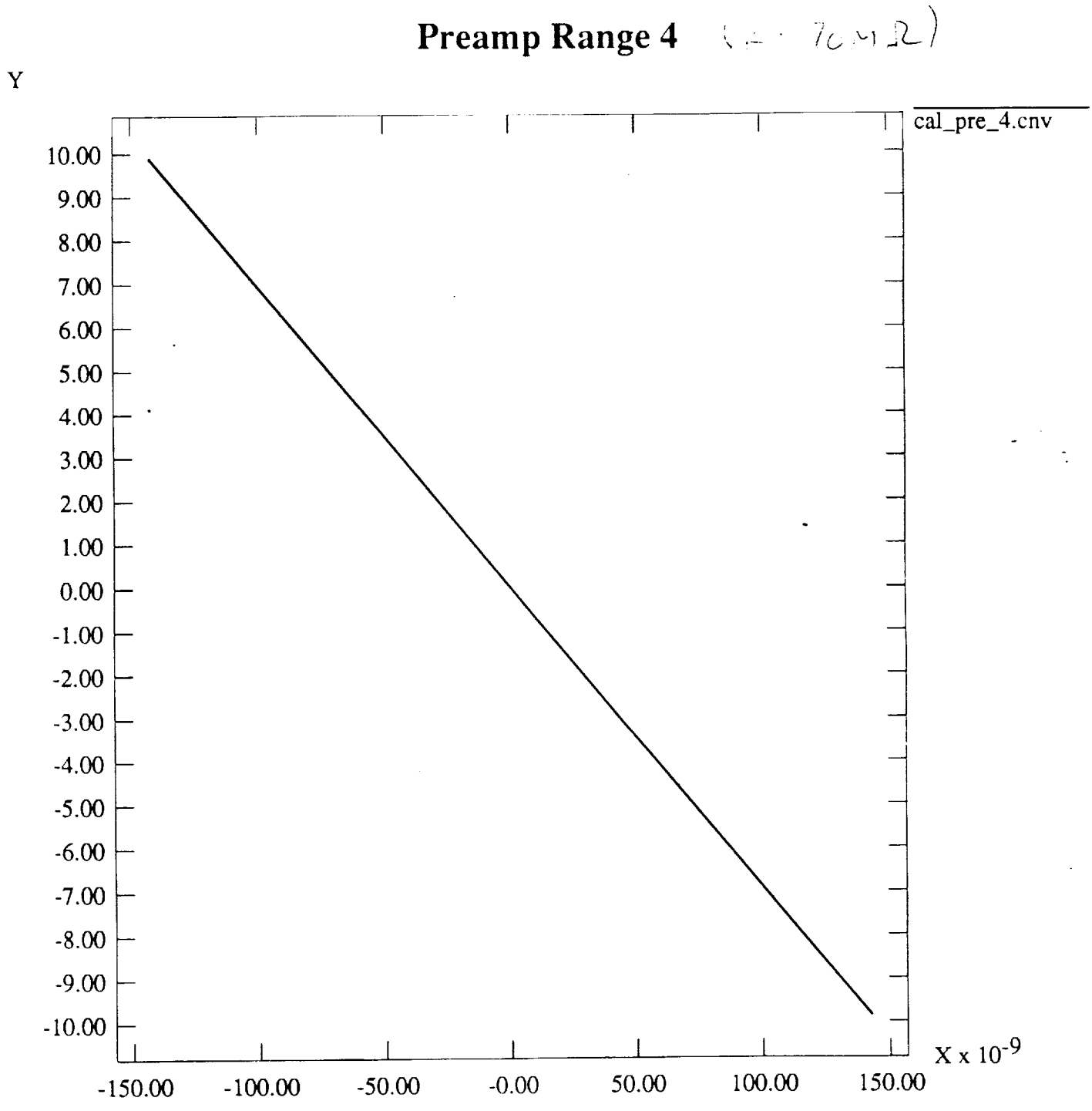


Figure A5. Preamp Range 4, Full Range (R=70M)

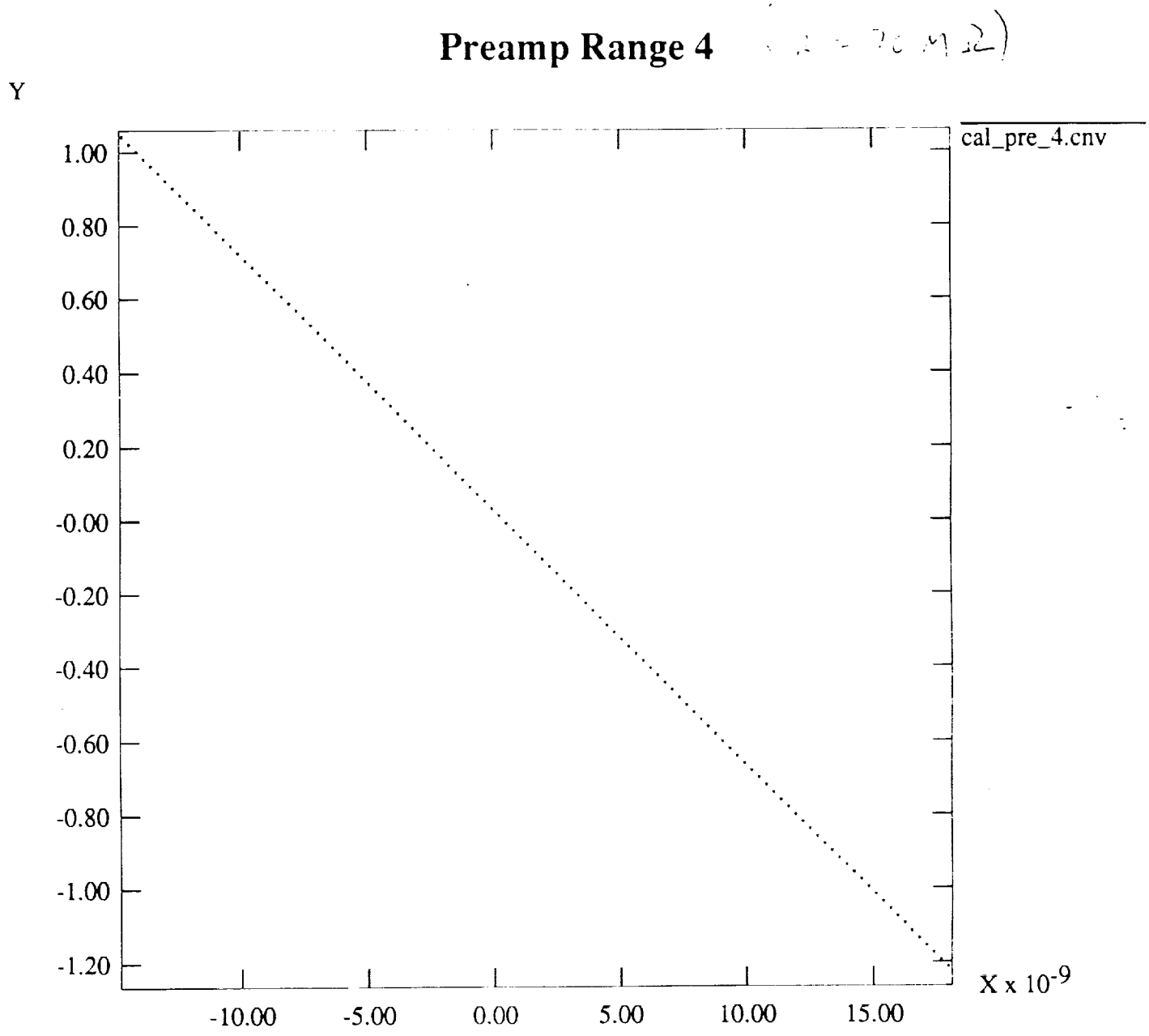


Figure A6. Preamp Range 4, Reduced Range (R=70M)

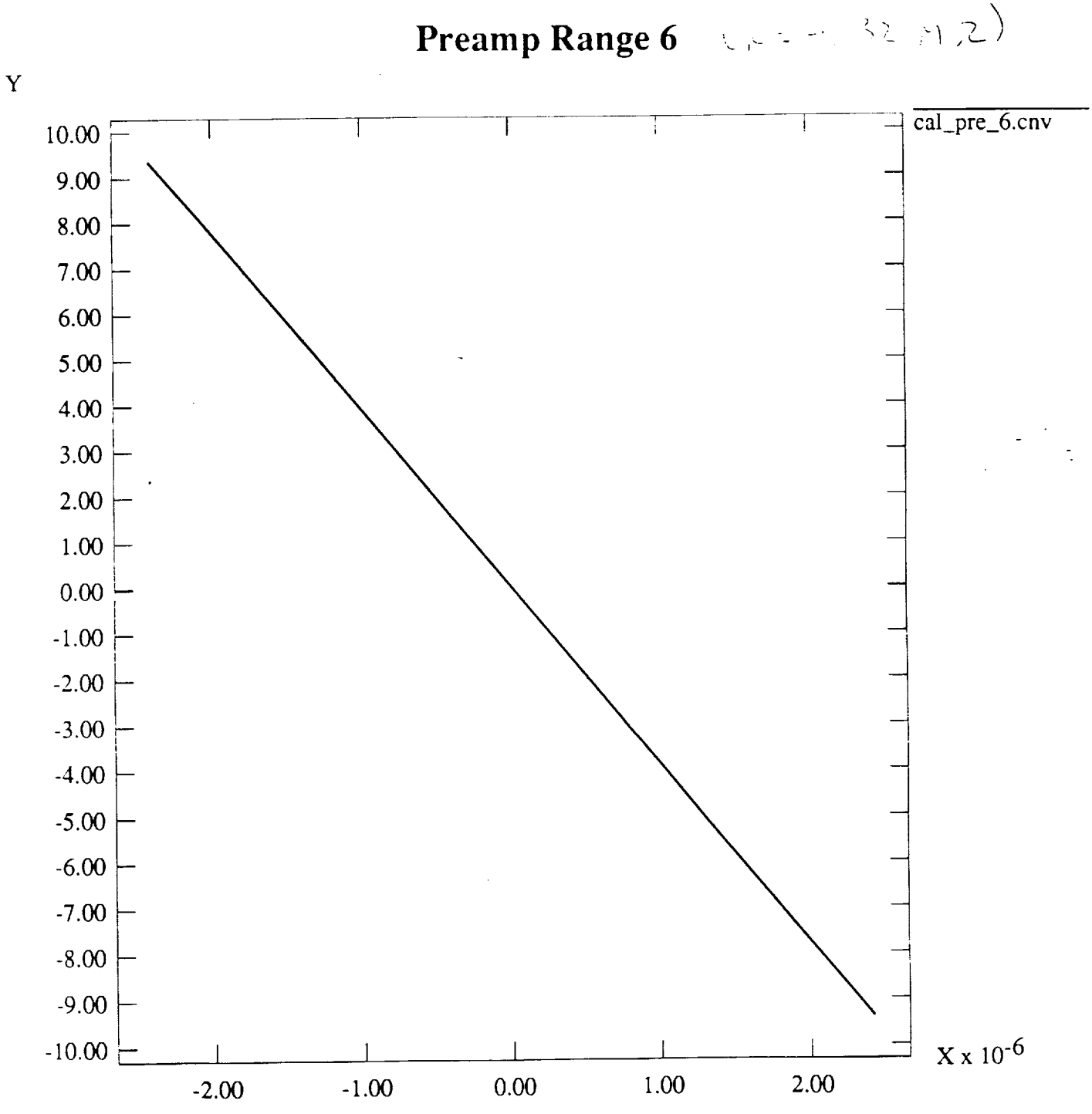


Figure A7. Preamp Range 6, Full Range (R=4.132M)

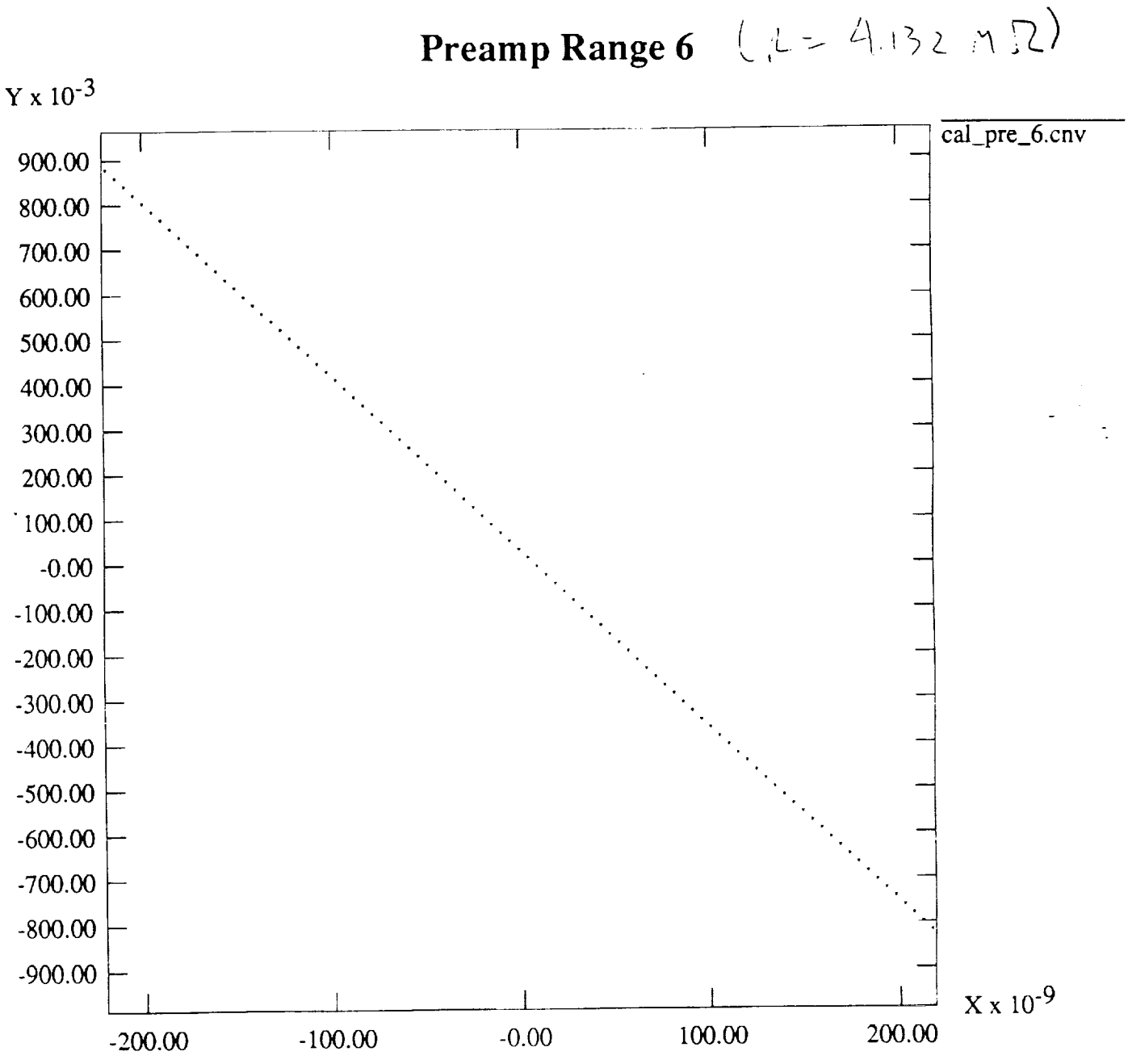


Figure A8. Preamp Range 6, Reduced Range (R=4.132M)

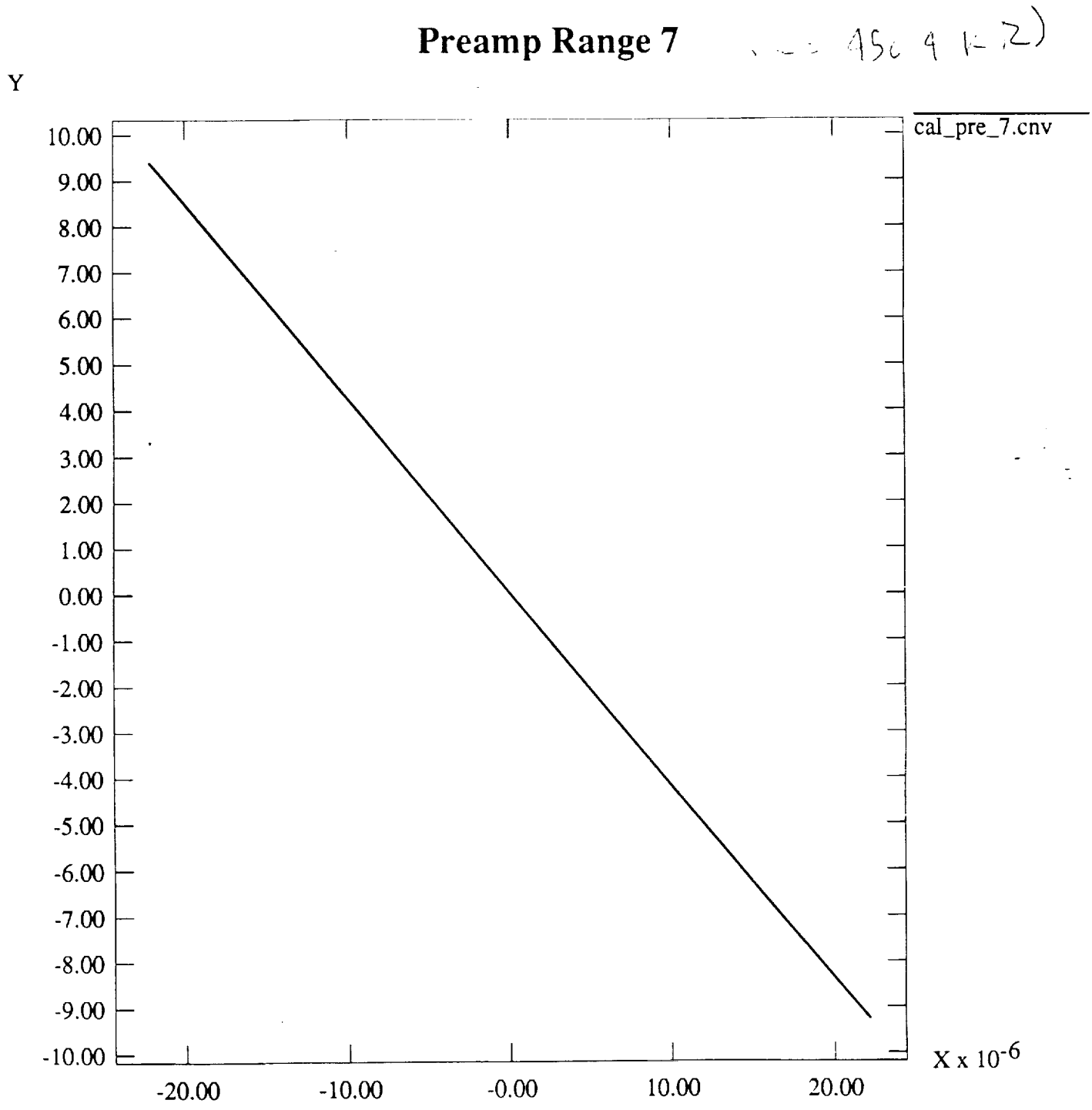


Figure A9. Preamp Range 7, Full Range (R=450.4K)